



# ***Institut Supérieur d'Informatique et de Mathématiques de Monastir***



***Niveau: ING1\_INF***

***Groupe:TD2***

***Réaliser par:***

***Laajili Oussama***

***&***

***Nsir Mohamed Akram***

***Enseignant: Mr Mahmoudi Ramzi***

***Année universitaire: 2023-2024***



## ***Table de matière:***

### ***Chapitre I: Environnement de développement JavaCard***

I.1 Préparation des outils logiciels nécessaires :

I.2 Instructions d'installation :

I.2.1 Installation d'Eclipse sous windows :

a. Installation du JDK:

b. Installation de l'IDE - Eclipse :

I.2.2 Installation du Java Card Development Kit 2 .2.2 :

I.2.3 Mise à jour des plugins Eclipse-JCDE :

### ***Chapitre II: Développement d'une application coté serveur***

I. Les différentes étapes de développement:

I.1 Programmation de l'application Serveur:

I.1.1 Création de l'applet card sous Eclipse :

1. Création d'un nouveau projet :

2. Création d'une applet Javacard:

I.1.2 Codage de notre applet :

1. Ajouter API JavaCard:

2. Création d'une applet Javacard:

3. Déclarer les attributs et les constantes :

4. Définition des méthodes publiques qu'elle doit obligatoirement implémenter :

I.1.3 Outils de simulation :

1. Installons notre applet

2. sélectionner notre applet

3. tester notre applet

4. déconnecter du simulateur

### ***Chapitre III: Programmation d'une application coté client***

I.1 Création de l'application client sous Eclipse :

1. Création d'un nouveau projet :

2. Ajout de la librairie « apduio » dans le classpath:

### 3.Création de la classe principale :

I.2 Utilisation de l'application cliente avec un simulateur – JCWDE:

1. créer un fichier "de configuration:

2. lancer notre simulateur:

3. lançons notre application cliente :

4. interroger le compteur :

5. Quittons maintenant notre application cliente:

### **Chapitre IV:** Mini projet:

I- Partie client:

I-1- préparation de l'interface graphique:

I-2- Les différents interfaces de notre application:

I-3- implémentation des méthodes nécessaires:

II- Partie serveur:

I-1- Déclaration des variables et des constantes:

I-2- Implémentations des méthodes nécessaires:

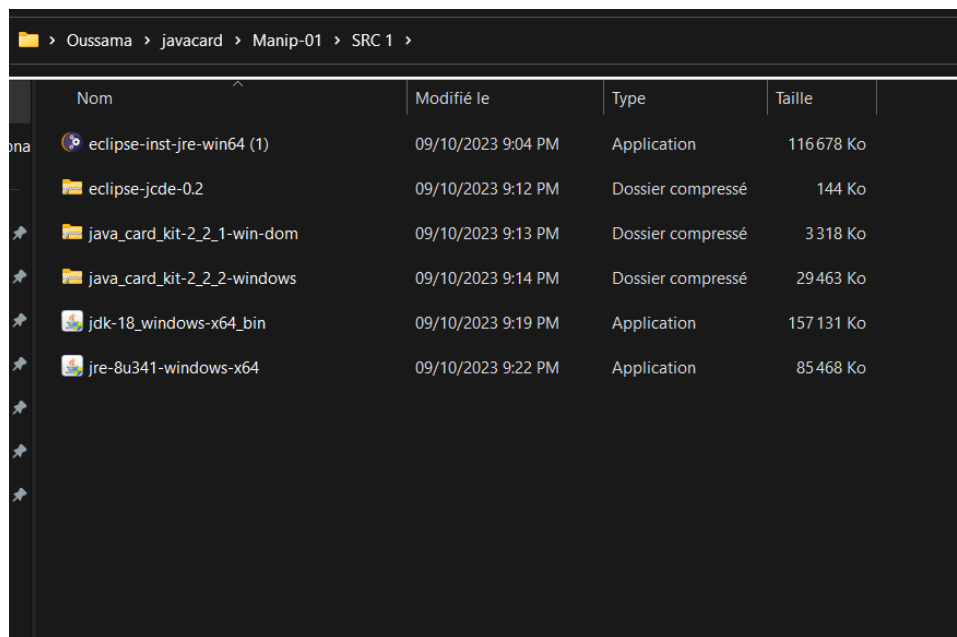
## **TP1 - Environnement de développement JavaCard**

---

### **But de ce TP :**

Ce TP a pour but d'installer l'environnement de développement nécessaire pour programmer des applications JavaCard. Nous allons installer Eclipse l'IDE (Environnement de développement intégré), la plate-forme JavaCard 2.2.2 (kit de développement) ainsi que le plug-in Eclipse-JCDE (interface entre la plate-forme JavaCard et Eclipse).

### **I.1 Préparation des outils logiciels nécessaires :**



The screenshot shows a Windows File Explorer window with the address bar displaying the path: > Oussama > javacard > Manip-01 > SRC 1 >. The main area displays a list of files and folders with columns for 'Nom', 'Modifié le', 'Type', and 'Taille'.







Nom	Modifié le	Type	Taille
eclipse-inst-jre-win64 (1)	09/10/2023 9:04 PM	Application	116 678 Ko
eclipse-jcde-0.2	09/10/2023 9:12 PM	Dossier compressé	144 Ko
java_card_kit-2_2_1-win-dom	09/10/2023 9:13 PM	Dossier compressé	3 318 Ko
java_card_kit-2_2_2-windows	09/10/2023 9:14 PM	Dossier compressé	29 463 Ko
jdk-18_windows-x64_bin	09/10/2023 9:19 PM	Application	157 131 Ko
jre-8u341-windows-x64	09/10/2023 9:22 PM	Application	85 468 Ko

### **I.2 Instructions d'installation :**

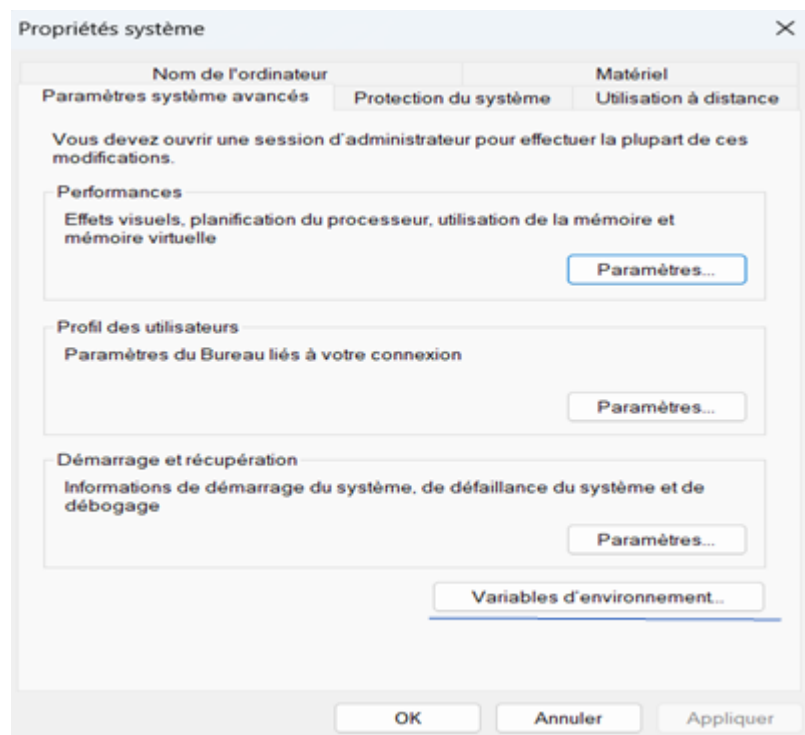
#### **I.2.1 Installation d'Eclipse sous windows :**

##### **a. Installation du JDK:**

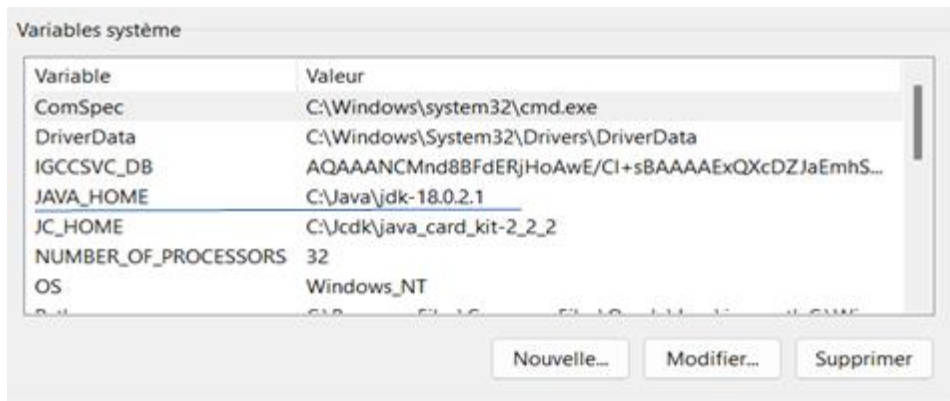
» Oussama » javacard » Manip-01 » SRC 1 »

Nom	Modifié le	Type	Taille
 eclipse-inst-jre-win64 (1)	09/10/2023 9:04 PM	Application	116 678 Ko
 eclipse-jcde-0.2	09/10/2023 9:12 PM	Dossier compressé	144 Ko
 java_card_kit-2_2_1-win-dom	09/10/2023 9:13 PM	Dossier compressé	3 318 Ko
 java_card_kit-2_2_2-windows	09/10/2023 9:14 PM	Dossier compressé	29 463 Ko
 jdk-18_windows-x64_bin	09/10/2023 9:19 PM	Application	157 131 Ko
 jre-8u341-windows-x64	09/10/2023 9:22 PM	Application	85 468 Ko

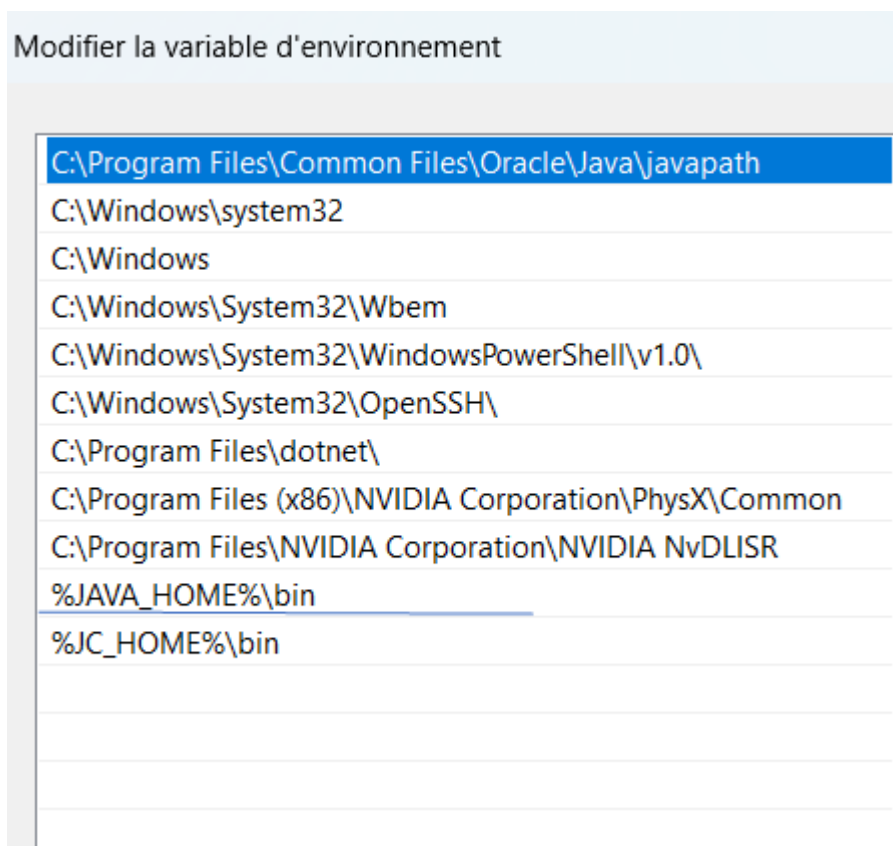
Accéder aux variables d'environnement:



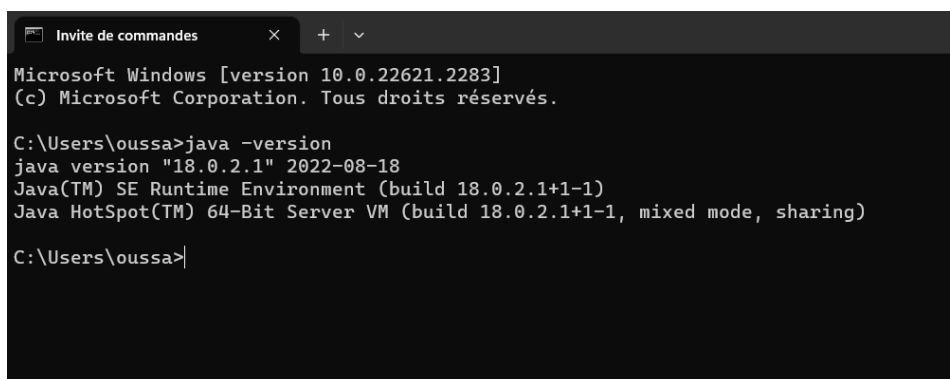
ajouter la variable d'environnement JAVA\_HOME



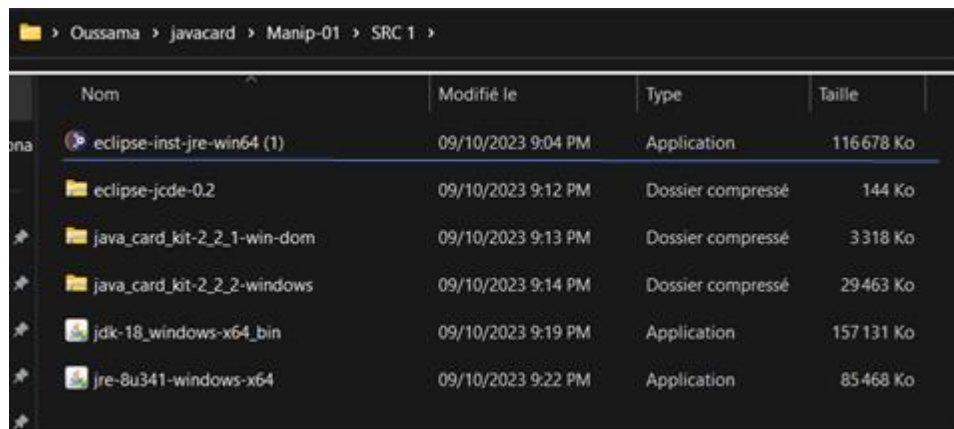
Modifier la variable Path en ajoutant vers sa fin : %JAVA\_HOME%\bin; :



Taper la commande « java -version » sous la console DOS :

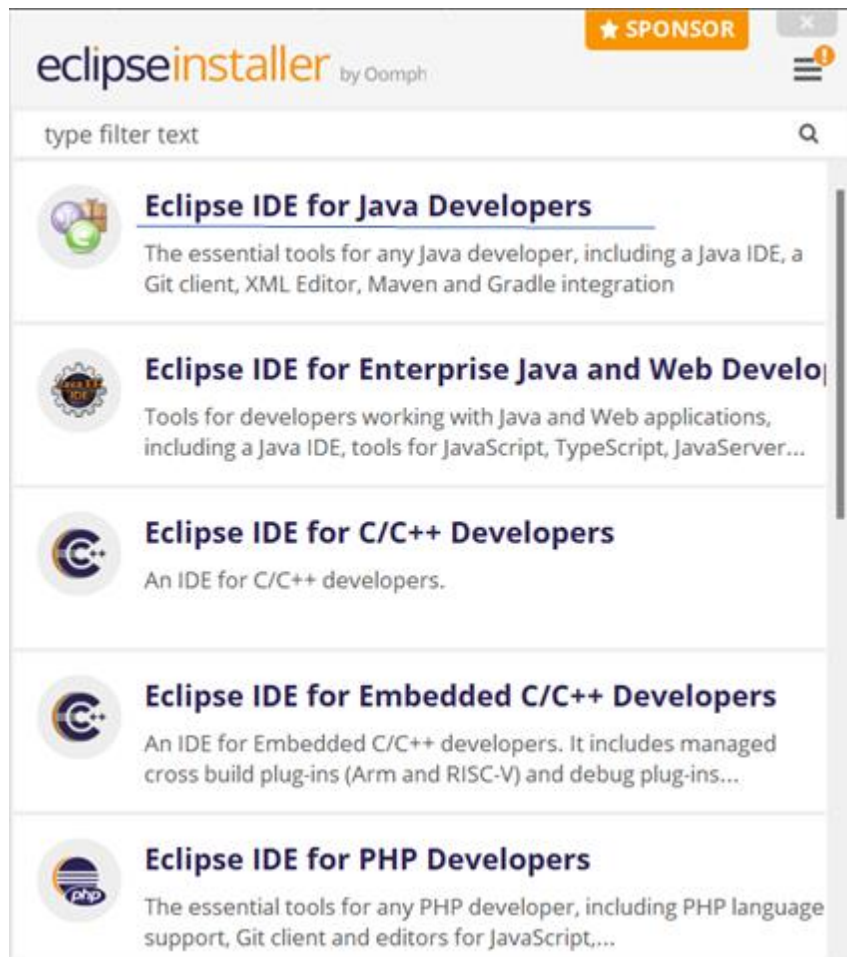


## b. Installation de l'IDE - Eclipse :



Nom	Modifié le	Type	Taille
eclipse-inst-jre-win64 (1)	09/10/2023 9:04 PM	Application	116 678 Ko
eclipse-jdt-0.2	09/10/2023 9:12 PM	Dossier compressé	144 Ko
java_card_kit-2_2_1-win-dom	09/10/2023 9:13 PM	Dossier compressé	3 318 Ko
java_card_kit-2_2_2-windows	09/10/2023 9:14 PM	Dossier compressé	29 463 Ko
jdk-18_windows-x64_bin	09/10/2023 9:19 PM	Application	157 131 Ko
jre-8u341-windows-x64	09/10/2023 9:22 PM	Application	85 468 Ko

Choisir Eclipse IDE for Java Developers :



Installation:





## Eclipse IDE for Java Developers

[details](#)

The essential tools for any Java developer, including a Java IDE, a Git client, XML Editor, Maven and Gradle integration.

Java 17+ VM	C:\Java\jdk-18.0.2.1	▼	📁
Installation Folder	C:\Users\oussa\eclipse\java-2022-09		📁

☒ create start menu entry

☒ create desktop shortcut

[📄 INSTALL](#)

Tester le bon fonctionnement d'Eclipse:

Comme initiation, nous allons créer une petite application Java avec l'IDE Eclipse qui va afficher à l'écran "Hello, World!":

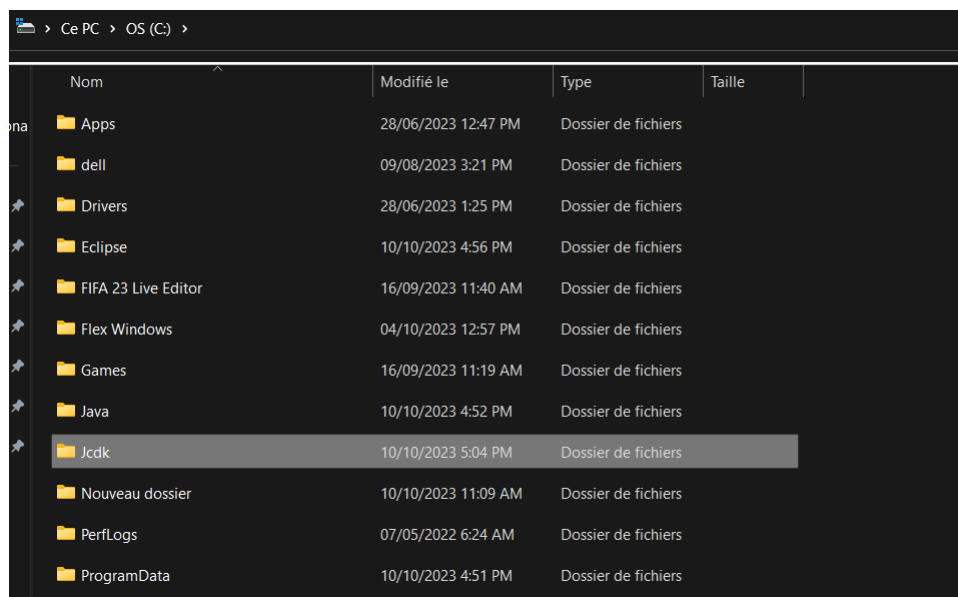
```
1 package aa;
2
3 public class aa {
4
5     public static void main(String[] args) {
6         System.out.print("Hello, World!");
7     }
8 }
9
10
11
```

Problems Javadoc Declaration Console ×

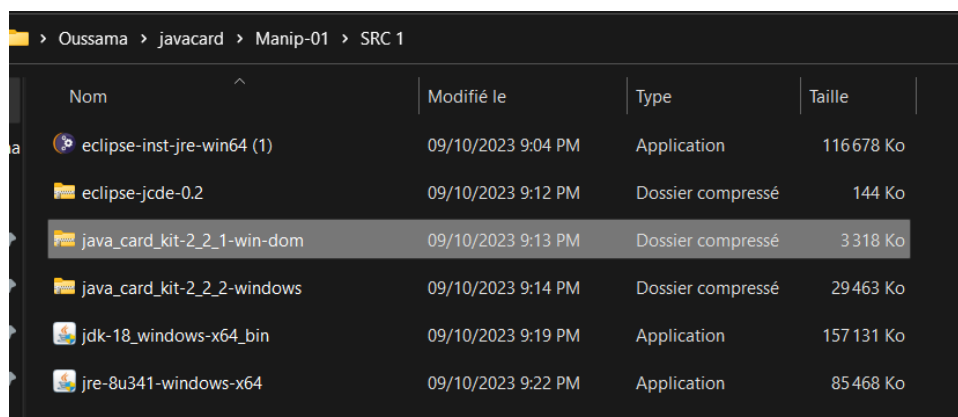
<terminated> aa [Java Application] C:\Java\jdk-18.0.2.1\bin\javaw.exe (11 oct. 2023, 12:0  
Hello, World!

## I.2.2 Installation du Java Card Development Kit 2.2.2 :

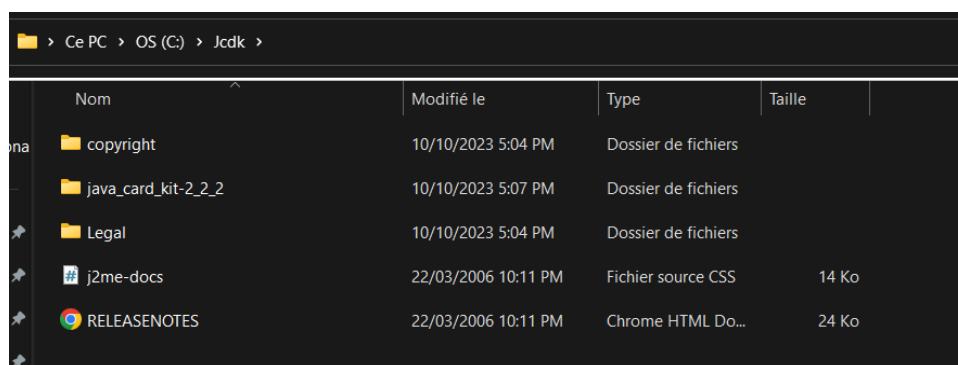
créer un nouveau répertoire C:\JCDK :



Décompresser l'archive java\_card\_kit-2\_2\_2-windows.zip après l'avoir téléchargé dans le répertoire C:\JCDK:



On va voir ceci dans le répertoire C:\JCDK:



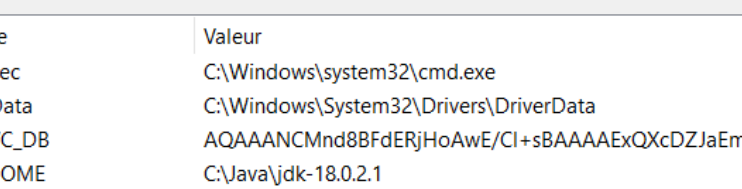
Décompresser toutes les fichiers du répertoire C:\JCDK  
 \java\_card\_kit-2\_2\_2 :

java_card_kit-2_2_2-rr-ant-tasks	22/03/2006 10:11 PM	Dossier compressé	810 Ko
java_card_kit-2_2_2-rr-bin-docs-do	22/03/2006 10:10 PM	Dossier compressé	3 445 Ko
java_card_kit-2_2_2-rr-bin-windows-do	22/03/2006 10:11 PM	Dossier compressé	1 842 Ko
java_card_kit-2_2_2-rr-specs	22/03/2006 10:11 PM	Dossier compressé	4 100 Ko

Pour avoir:

Nom	Modifié le	Type	Taille
ant-tasks	10/10/2023 5:06 PM	Dossier de fichiers	
api_export_files	10/10/2023 5:07 PM	Dossier de fichiers	
bin	10/10/2023 5:07 PM	Dossier de fichiers	
doc	10/10/2023 5:06 PM	Dossier de fichiers	
jc_specification	10/10/2023 5:07 PM	Dossier de fichiers	
lib	10/10/2023 5:07 PM	Dossier de fichiers	
samples	10/10/2023 5:07 PM	Dossier de fichiers	
java_card_kit-2_2_2-rr-ant-tasks	22/03/2006 10:11 PM	Dossier compressé	810 Ko
java_card_kit-2_2_2-rr-bin-docs-do	22/03/2006 10:10 PM	Dossier compressé	3 445 Ko
java_card_kit-2_2_2-rr-bin-windows-do	22/03/2006 10:11 PM	Dossier compressé	1 842 Ko
java_card_kit-2_2_2-rr-specs	22/03/2006 10:11 PM	Dossier compressé	4 100 Ko

Créer une nouvelle variable d'environnement JC\_HOME contenant le chemin C:\JCDK\java\_card\_kit-2\_2\_2:



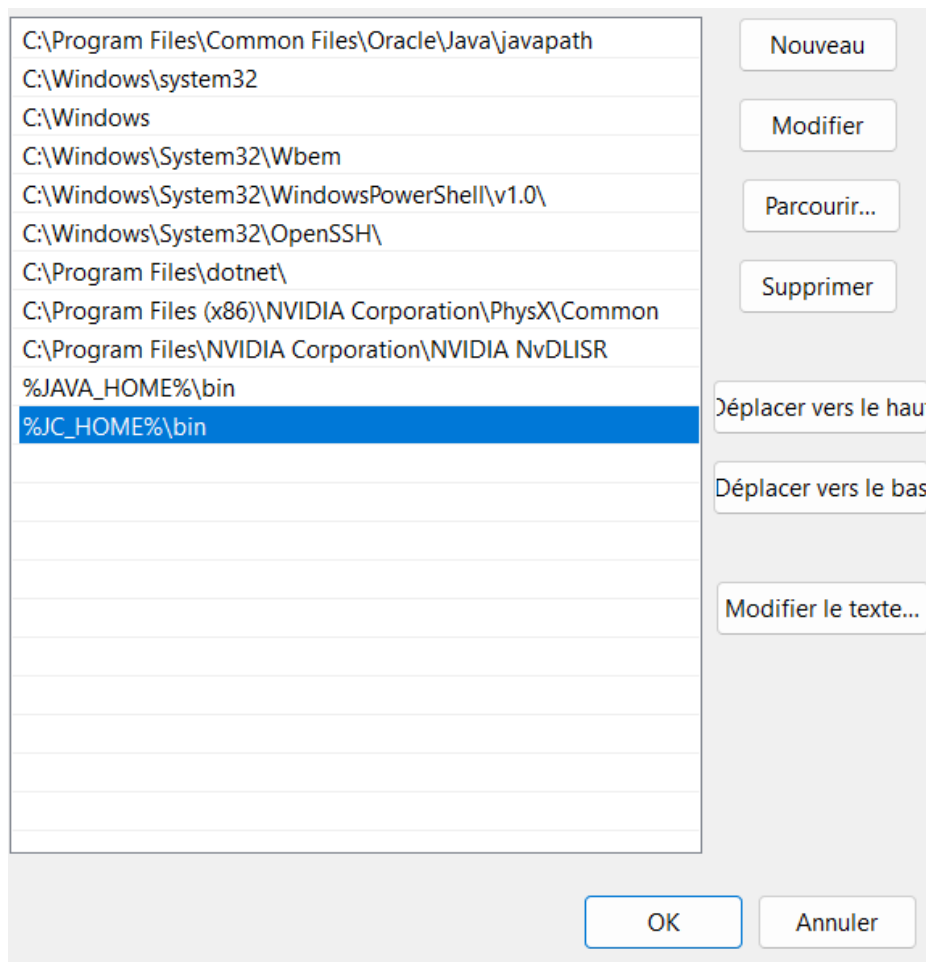
Variables système

Variable	Valeur
ComSpec	C:\Windows\system32\cmd.exe
DriverData	C:\Windows\System32\Drivers\DriverData
IGCCSVC_DB	AQAAANCmnd8BFdERjHoAwE/CI+sBAAAAExQXcDZJaEmhS...
JAVA_HOME	C:\Java\jdk-18.0.2.1
JC_HOME	C:\Jcdk\java_card_kit-2_2_2
NUMBER_OF_PROCESSORS	32
OS	Windows_NT

Nouvelle... Modifier... Supprimer

OK Annuler

Modifier la variable Path : %JC\_HOME%\bin:



taper « apdutool » dans une console de commandes :

```
Invite de commandes
Microsoft Windows [version 10.0.22621.2283]
(c) Microsoft Corporation. Tous droits réservés.

C:\Users\oussa>apdutool
Java Card 2.2.2 APDU Tool, Version 1.3
Copyright 2005 Sun Microsystems, Inc. All rights reserved. Use is subject to license terms.
Opening connection to localhost on port 9025.
java.net.ConnectException: Connection refused: connect

C:\Users\oussa>
```

### I.2.3 Mise à jour des plugins Eclipse-JCDE :

décompresser l'archive eclipse-jcde-0.2.zip après l'avoir téléchargée dans le répertoire C:\Eclipse\eclipse\plugins:

📁 > Oussama > javacard > Manip-01 > SRC 1

Nom	Modifié le	Type	Taille
eclipse-inst-jre-win64 (1)	09/10/2023 9:04 PM	Application	116 678 Ko
eclipse-jcde-0.2	09/10/2023 9:12 PM	Dossier compressé	144 Ko
java_card_kit-2_2_1-win-dom	09/10/2023 9:13 PM	Dossier compressé	3 318 Ko
java_card_kit-2_2_2-windows	09/10/2023 9:14 PM	Dossier compressé	29 463 Ko
jdk-18_windows-x64_bin	09/10/2023 9:19 PM	Application	157 131 Ko
jre-8u341-windows-x64	09/10/2023 9:22 PM	Application	85 468 Ko

📁 > Ce PC > OS (C:) > Eclipse > eclipse > plugins >

Nom	Modifié le	Type	Taille
_MACOSX	10/10/2023 5:12 PM	Dossier de fichiers	
plugins	10/10/2023 5:12 PM	Dossier de fichiers	
.DS_Store	31/07/2012 10:31 AM	Fichier DS_STORE	7 Ko

recopier tout le contenu de répertoire plugins directement sous C:\Eclipse\eclipse\plugins:

📁 > Ce PC > OS (C:) > Eclipse > eclipse > plugins > plugins

Nom	Modifié le	Type	Taille
.DS_Store	31/07/2012 10:31 AM	Fichier DS_STORE	7 Ko
org.eclipse.jcde.core_0.2.0	31/07/2012 10:26 AM	Executable Jar File	48 Kc
org.eclipse.jcde.cref_0.1.0	21/09/2006 11:38 PM	Executable Jar File	9 Kc
org.eclipse.jcde.jcbp.wizards_0.1.0	21/09/2006 11:38 PM	Executable Jar File	9 Kc
org.eclipse.jcde.jcbp_0.1.0	21/09/2006 11:38 PM	Executable Jar File	22 Kc
org.eclipse.jcde.jctools_0.1.0	21/09/2006 11:38 PM	Executable Jar File	21 Kc
org.eclipse.jcde.jcwde_0.1.0	21/09/2006 11:38 PM	Executable Jar File	10 Kc
org.eclipse.jcde.preferences_0.1.0	21/09/2006 11:38 PM	Executable Jar File	8 Ko
org.eclipse.jcde.wizards_0.1.0	21/09/2006 11:38 PM	Executable Jar File	34 Ko

Copier (Ctrl+C)

Ajouter aux Favoris

Compresser dans un fichier ZIP

Copier en tant que chemin d'accès Ctrl+Maj+C

Propriétés Alt+Entrée

Afficher d'autres d'options

1

# **Développement d'une application coté serveur 2.0**

## **But de ce TP :**

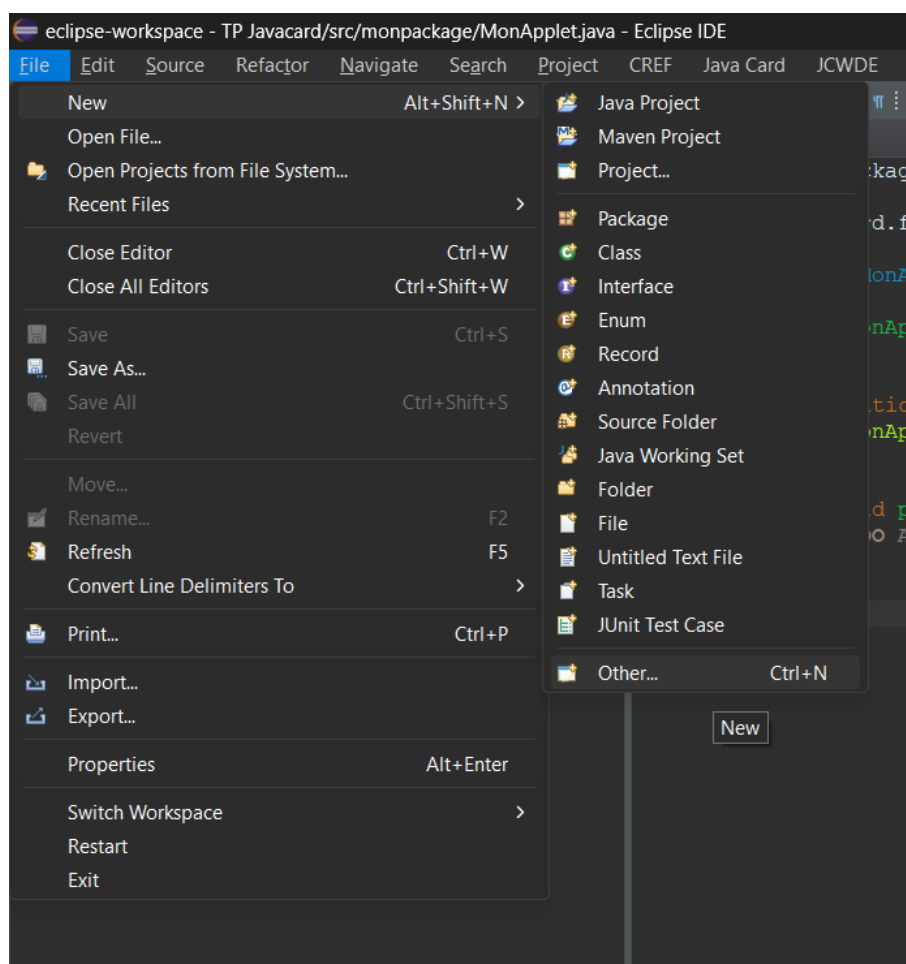
Ce TP a pour but de développer une première applet Java Card (appelée parfois cardlet car elle s'exécute sur la carte à puce), et de les tester à l'aide d'un simulateur de carte.

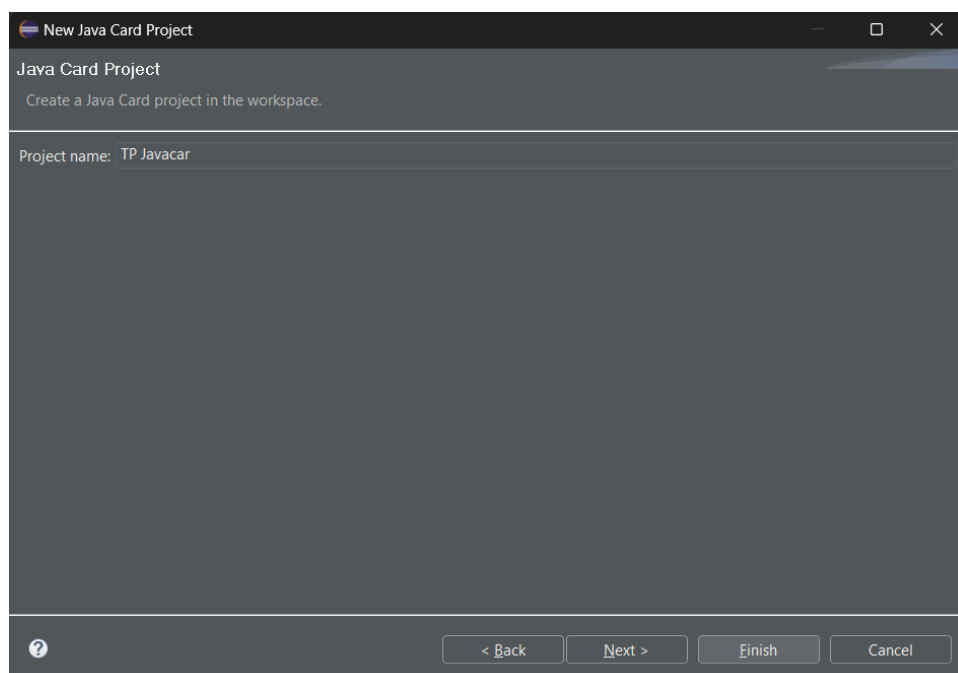
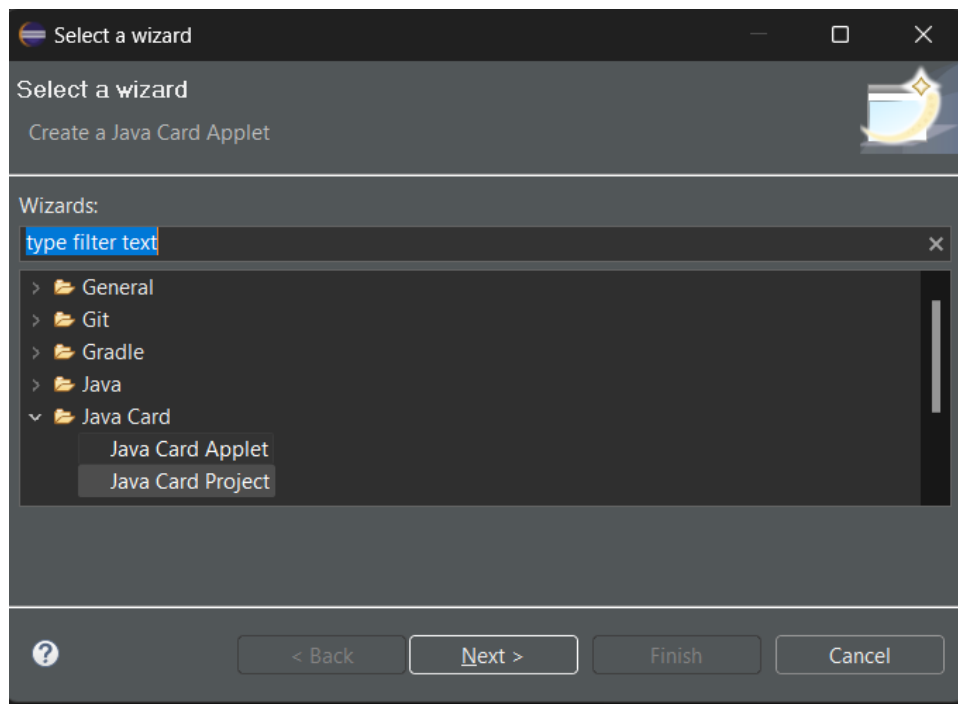
## I. Les différentes étapes de développement:

### I.1 Programmation de l'application Serveur:

#### I.1.1 Création de l'applet card sous Eclipse :

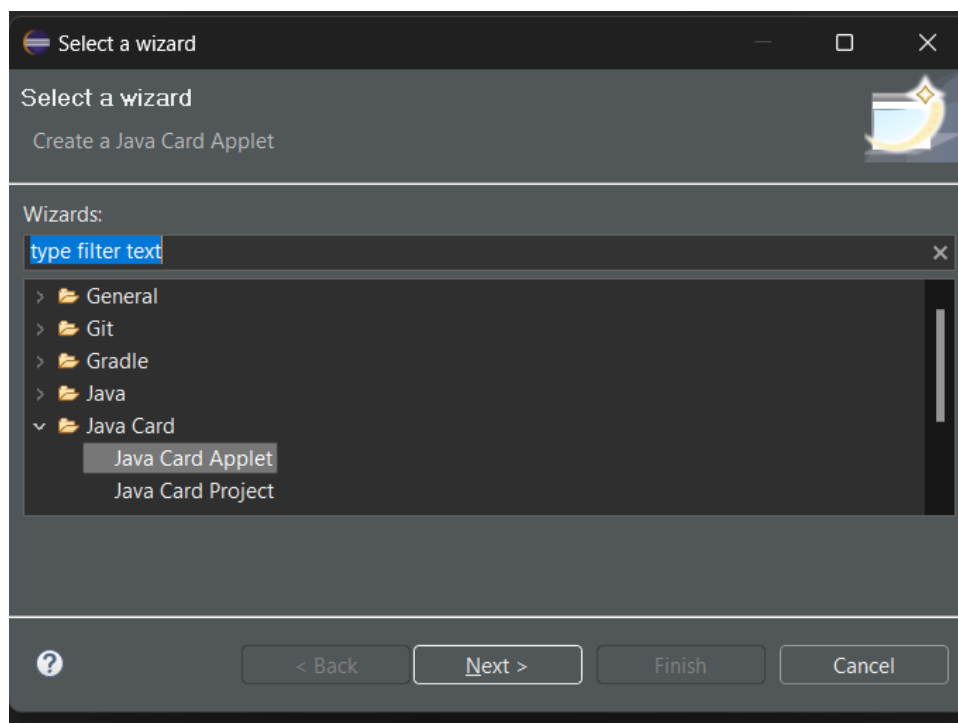
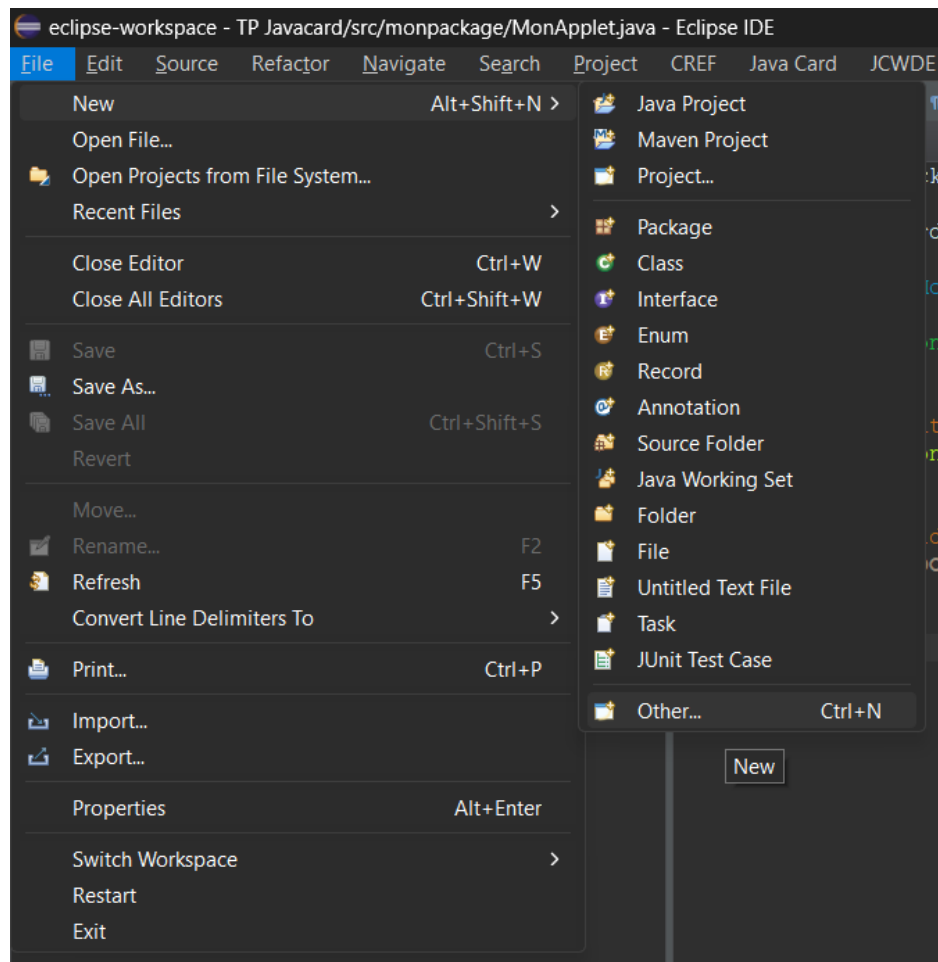
##### 1. Création d'un nouveau projet :







## 2. Création d'une applet Javacard:



**New Java Card Applet**

Java Card Applet

⚠ Type name is discouraged. By convention, Java type names usually start with an uppercase letter

Source folder: TP Javacard/src Browse...

Package: monpackage Browse...

☐ Enclosing type: monpackage.MonApplet Browse...

---

Applet AID: 0x01:0x02:0x03:0x04:0x05:0x06:0x07:0x08:0x09:0x00:0x00

Name: monapple

Modifiers: ☒ public ☐ package ☐ private ☐ protected  
☐ abstract ☐ final ☐ static  
☒ none ☐ sealed ☐ non-sealed ☐ final

Superclass: javacard.framework.Applet Browse...

Interfaces: Add... Remove

Do you want to add comments? (Configure templates and default value [here](#))  
☐ Generate comments

? < Back Next > Finish Cancel

## I.1.2 Codage de notre applet :

On peut séparer l'écriture de notre applet Java Card en plusieurs étapes :

Étape 1. Ajouter API JavaCard :

```
*MonApplet.java x
1 package monpackage;
2
3 import javacard.framework.APDU;
4 import javacard.framework.Applet;
5 import javacard.framework.ISOException;
6 import javacard.framework.ISO7816;
7
```

Étape 2. Déclarer les attributs et les constantes :

```
8 public class MonApplet extends Applet {
9     public static final byte CLA_MONAPPLET = (byte) 0xB0;
10    public static final byte INS_INCREMENTER_COMPTEUR = 0x00;
11    public static final byte INS_DECREMENTER_COMPTEUR = 0x01;
12    public static final byte INS_INTERROGER_COMPTEUR = 0x02;
13    public static final byte INS_INITIALISER_COMPTEUR = 0x03;
14
15    private byte compteur;
16
```

Étape 3. Définition des méthodes publiques qu'elle doit obligatoirement implémenter :

1) la méthode Install () :

```
14 private byte compteur;
15
16
17 private MonApplet() {
18     compteur = 0;
19 }
20
21 public static void install(byte bArray[], short bOffset, byte bLength) throws ISOException {
22     new MonApplet().register();
23 }
24
```

2) la méthode process () :

a. Extraire le buffer APDU :

```
public void process(APDU apdu) throws ISOException {
    byte[] buffer = apdu.getBuffer();
```

b. Vérifier les octets entête d'APDU :

```
public void process(APDU apdu) throws ISOException {
    byte[] buffer = apdu.getBuffer();

    if (this.selectingApplet()) return;
    if (buffer[ISO7816.OFFSET_CLA] != CLA_MONAPPLET) {
        ISOException.throwIt(ISO7816.SW_CLA_NOT_SUPPORTED);
    }
}
```

c. Traiter la commande APDU :

```
public void process(APDU apdu) throws ISOException {
    byte[] buffer = apdu.getBuffer();

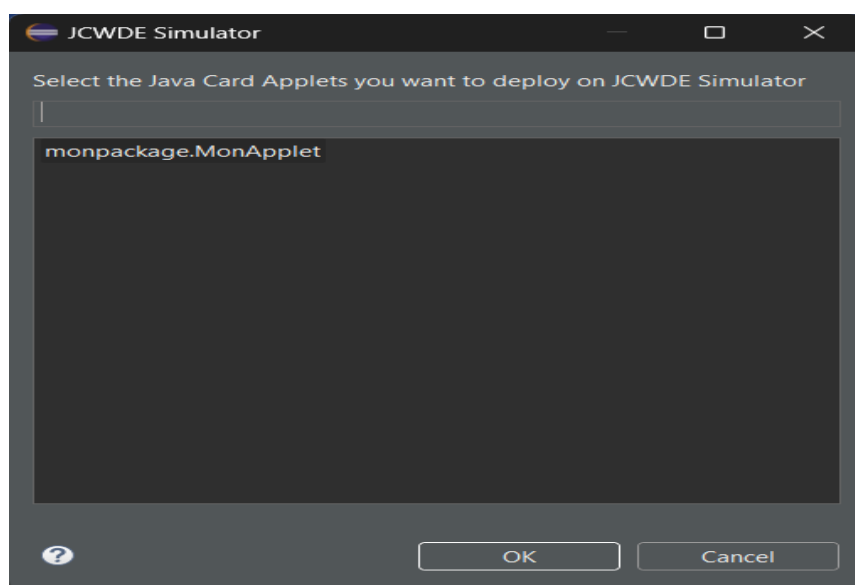
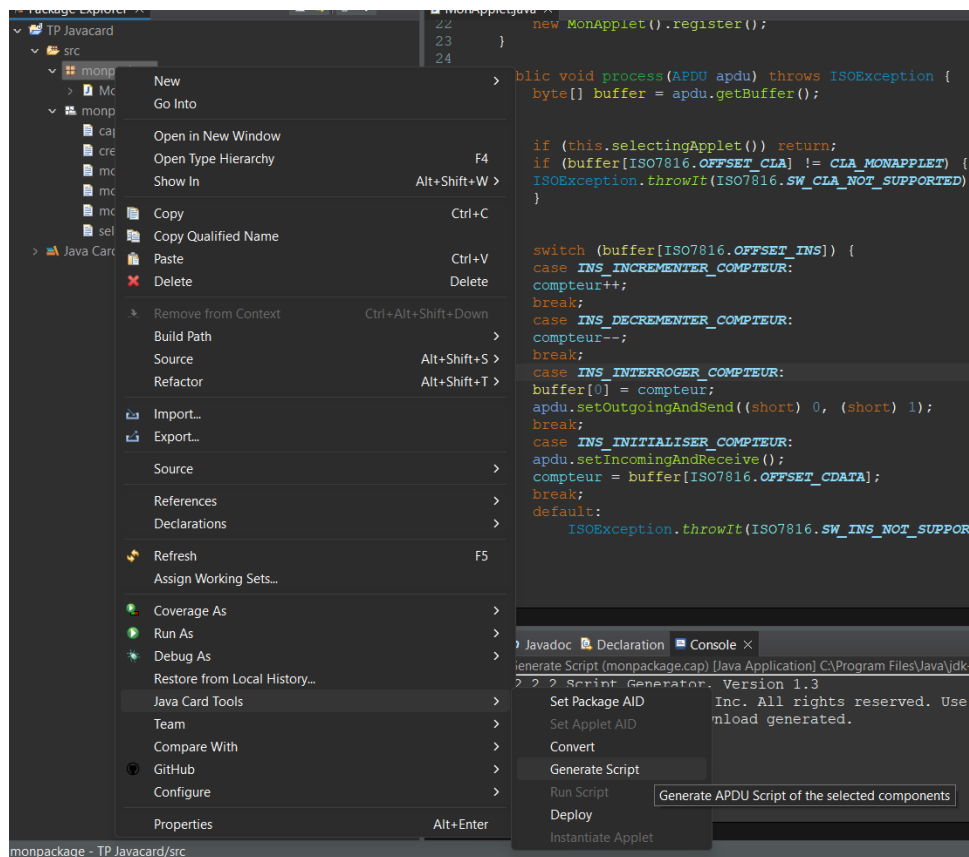
    if (this.selectingApplet()) return;
    if (buffer[ISO7816.OFFSET_CLA] != CLA_MONAPPLET) {
        ISOException.throwIt(ISO7816.SW_CLA_NOT_SUPPORTED);
    }

    switch (buffer[ISO7816.OFFSET_INS]) {
        case INS_INCREMENTER_COMPTEUR:
            compteur++;
            break;
        case INS_DECREMENTER_COMPTEUR:
            compteur--;
            break;
        case INS_INTERROGER_COMPTEUR:
            buffer[0] = compteur;
            apdu.setOutgoingAndSend((short) 0, (short) 1);
            break;
        case INS_INITIALISER_COMPTEUR:
            apdu.setIncomingAndReceive();
            compteur = buffer[ISO7816.OFFSET_CDATA];
            break;
        default:
    }
}
```

d. Renvoyer le mot d'état (Word status):

```
default:
    ISOException.throwIt(ISO7816.SW_INS_NOT_SUPPORTED);
}
```

### I.1.3 Outils de simulation :



## Installons notre applet

```
Invite de commandes - apduTool x + v
Microsoft Windows [version 10.0.22621.2428]
(c) Microsoft Corporation. Tous droits réservés.

C:\Users\oussa>cd c:\

c:\>apdutool
Java Card 2.2.2 APDU Tool, Version 1.3
Copyright 2005 Sun Microsystems, Inc. All rights reserved. Use is subject to license terms.
Opening connection to localhost on port 9025.
Connected.
powerup;
// Select the installer applet
0x00 0xA4 0x04 0x00 0x00 0xA0 0x00 0x00 0x62 0x03 0x01 0x08 0x01 0x7F;
// create MonApplet applet
Received ATR = 0x3b 0xf0 0x11 0x00 0xff 0x00
CLA: 00, INS: a4, P1: 04, P2: 00, Lc: 09, a0, 00, 00, 00, 62, 03, 01, 08, 01, Le: 00, SW1: 90, SW2: 00
0x00 0xB8 0x00 0x00 0xd 0xb 0x01 0x02 0x03 0x04 0x05 0x06 0x07 0x08 0x09 0x00 0x00 0x7F;
CLA: 80, INS: b8, P1: 00, P2: 00, Lc: 0d, 0b, 01, 02, 03, 04, 05, 06, 07, 08, 09, 00, 00, 00, Le: 0b, 01, 02, 03, 04, 05, 06, 07, 08, 09, 00, 00, SW1: 90, SW2: 00
|
```

## sélectionner notre applet

```
powerup;
// select MonApplet applet
Received ATR = 0x3b 0xf0 0x11 0x00 0xff 0x00
0x00 0xA4 0x04 0x00 0xb 0x01 0x02 0x03 0x04 0x05 0x06 0x07 0x08 0x09 0x00 0x00 0x7F;
CLA: 00, INS: a4, P1: 04, P2: 00, Lc: 0b, 01, 02, 03, 04, 05, 06, 07, 08, 09, 00, 00, Le: 00, SW1: 90, SW2: 00
|
```

## tester notre applet

```
0xB0 0x02 0x00 0x00 0x00 0x7F;
CLA: b0, INS: 02, P1: 00, P2: 00, Lc: 00, Le: 01, 00, SW1: 90, SW2: 00
|
```

## incrémenter le compteur

```
0xB0 0x00 0x00 0x00 0x00 0x7F;
CLA: b0, INS: 00, P1: 00, P2: 00, Lc: 00, Le: 00, SW1: 90, SW2: 00
|
```

## Interrogeons de nouveau le compteur

```
0xB0 0x02 0x00 0x00 0x00 0x7F;
CLA: b0, INS: 02, P1: 00, P2: 00, Lc: 00, Le: 01, 01, SW1: 90, SW2: 00
|
```

## Initialisons maintenant (INS = 0x03)

```
0xB0 0x03 0x00 0x00 0x01 0x4A 0x7F;
CLA: b0, INS: 03, P1: 00, P2: 00, Lc: 01, 4a, Le: 00, SW1: 90, SW2: 00
|
```

## Décrémentons le compteur (INS = 0x01)

```
0xB0 0x01 0x00 0x00 0x00 0x7F;
CLA: b0, INS: 01, P1: 00, P2: 00, Lc: 00, Le: 00, SW1: 90, SW2: 00
|
```

## Interrogeons de nouveau le compteur (INS = 0x02)

```
0xB0 0x02 0x00 0x00 0x00 0x7F;
CLA: b0, INS: 02, P1: 00, P2: 00, Lc: 00, Le: 01, 49, SW1: 90, SW2: 00
|
```

déconnecter du simulateur

```
powerdown;
```

```
Problems Javadoc Declaration Console ×
<terminated> JCWDE [Java Application] C:\Program Files\Java\jdk-21\bin\javaw.exe (24 oct. 2023, 10:45:30 AM - 11:01:54 AM) [pid: 34404]
Java Card 2.2.2 Workstation Development Environment, Version 1.3
Copyright 2005 Sun Microsystems, Inc. All rights reserved. Use is subject to license terms.
jcwde is listening for T=1 Adu's on TCP/IP port 90025.
jcwde exiting on receipt of power down command.
```

c) CREF : simulateur avec conservation d'état:

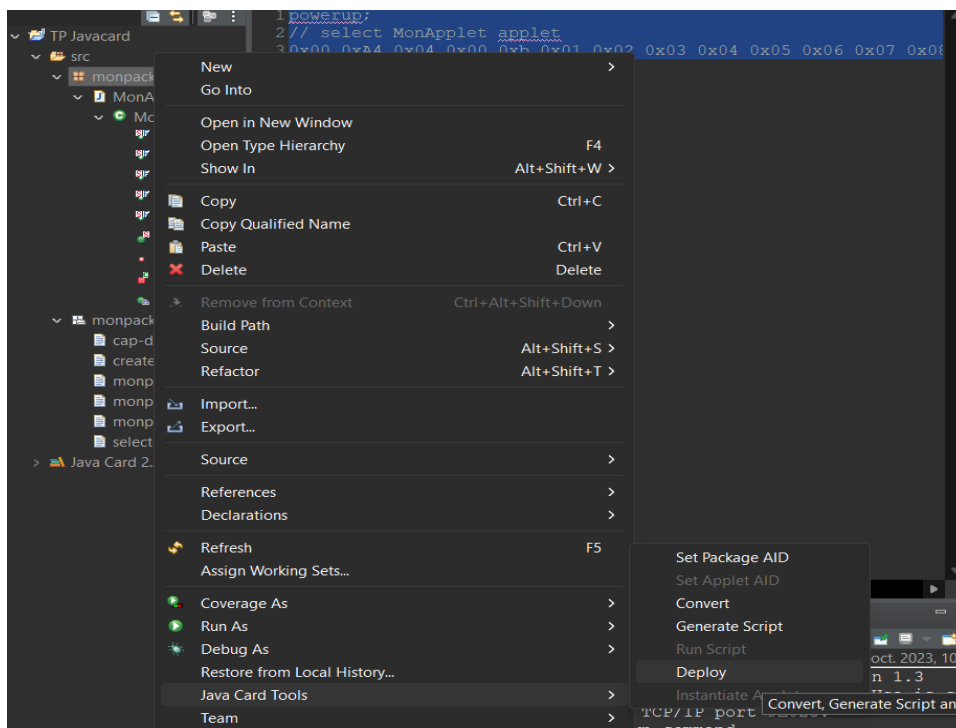
lancer CREF

```
C:\Users\oussa>cref -o monapplet.eeprom
Java Card 2.2.2 C Reference Implementation Simulator (version 0.41)
32-bit Address Space implementation - with cryptography support
T=1 / T=CL Dual interface APDU protocol (ISO 7816-3)
Copyright 2005 Sun Microsystems, Inc. All rights reserved.

Memory configuration
  Type      Base      Size      Max Addr
  RAM       0x0        0x1000    0xffff
  ROM       0x2000    0xe000    0xffff
  E2P       0x10020    0xffe0    0x1ffff

  ROM Mask size =                0xce64 =        52836 bytes
  Highest ROM address in mask =    0xee63 =        61027 bytes
  Space available in ROM =         0x119c =         4508 bytes
EEPROM will be saved in file "monapplet.eeprom"
Mask has now been initialized for use
```

Uploadons maintenant notre applet



```

Opening connection to localhost on port 9025.
Connected.
Received ATR = 0x3b 0xf0 0x11 0x00 0xff 0x01
CIA: 80, INS: a4, P1: 04, P2: 00, Lc: 09, a0, 00, 00, 00, 62, 03, 01, 08, 01, Le: 00, SW1: 90, SW2: 00
CIA: 80, INS: b0, P1: 00, P2: 00, Lc: 00, Le: 00, SW1: 90, SW2: 00
CIA: 80, INS: b2, P1: 01, P2: 00, Lc: 00, Le: 00, SW1: 90, SW2: 00
CIA: 80, INS: b4, P1: 01, P2: 00, Lc: 17, 01, 00, 14, de, ca, ff, ed, 01, 02, 04, 00, 01, 0a, 01, 02, 03, 04, 05, 06, 07, 08, 09, 00, Le: 00, SW1: 90, SW2: 00
CIA: 80, INS: bc, P1: 01, P2: 00, Lc: 00, Le: 00, SW1: 90, SW2: 00
CIA: 80, INS: b2, P1: 02, P2: 00, Lc: 00, Le: 00, SW1: 90, SW2: 00
CIA: 80, INS: b4, P1: 02, P2: 00, Lc: 20, 02, 00, 1f, 00, 14, 00, 1f, 00, 0f, 00, 15, 00, 2a, 00, 0c, 00, 7d, 00, 0a, 00, 15, 00, 00, 00, 61, 00, 00, 00, 00, 00, 02, Le: 00, SW1: 90, SW2: 00
CIA: 80, INS: b4, P1: 02, P2: 00, Lc: 02, 01, 00, Le: 00, SW1: 90, SW2: 00
CIA: 80, INS: bc, P1: 02, P2: 00, Lc: 00, Le: 00, SW1: 90, SW2: 00
CIA: 80, INS: b2, P1: 04, P2: 00, Lc: 00, Le: 00, SW1: 90, SW2: 00
CIA: 80, INS: b4, P1: 04, P2: 00, Lc: 18, 04, 00, 15, 02, 03, 01, 07, a0, 00, 00, 00, 62, 01, 01, 00, 01, 07, a0, 00, 00, 00, 62, 00, 01, Le: 00, SW1: 90, SW2: 00
CIA: 80, INS: bc, P1: 04, P2: 00, Lc: 00, Le: 00, SW1: 90, SW2: 00
CIA: 80, INS: b2, P1: 03, P2: 00, Lc: 00, Le: 00, SW1: 90, SW2: 00
CIA: 80, INS: b4, P1: 03, P2: 00, Lc: 12, 03, 00, 0f, 01, 0b, 01, 02, 03, 04, 05, 06, 07, 08, 09, 00, 00, 00, 0c, Le: 00, SW1: 90, SW2: 00
CIA: 80, INS: bc, P1: 03, P2: 00, Lc: 00, Le: 00, SW1: 90, SW2: 00
CIA: 80, INS: b2, P1: 06, P2: 00, Lc: 00, Le: 00, SW1: 90, SW2: 00
CIA: 80, INS: b4, P1: 06, P2: 00, Lc: 0f, 06, 00, 0c, 00, 80, 03, 01, ff, 00, 07, 01, 00, 00, 00, 19, Le: 00, SW1: 90, SW2: 00
CIA: 80, INS: bc, P1: 06, P2: 00, Lc: 00, Le: 00, SW1: 90, SW2: 00
CIA: 80, INS: b2, P1: 07, P2: 00, Lc: 00, Le: 00, SW1: 90, SW2: 00
CIA: 80, INS: b4, P1: 07, P2: 00, Lc: 20, 07, 00, 7d, 00, 02, 10, 18, 8c, 00, 01, 18, 03, 88, 00, 7a, 02, 30, 8f, 00, 02, 3d, 8c, 00, 03, 8b, 00, 04, 7a, 03, 21, 19, 8b, Le: 00, SW1: 90, SW2: 00
CIA: 80, INS: b4, P1: 07, P2: 00, Lc: 20, 00, 05, 2d, 18, 8b, 00, 06, 60, 03, 7a, 1a, 03, 25, 10, b0, 6a, 08, 11, 6e, 00, 8d, 00, 07, 1a, 04, 25, 73, 00, 3f, 00, 00, 00, Le: 00, SW1: 90, SW2: 00
CIA: 80, INS: b4, P1: 07, P2: 00, Lc: 20, 03, 00, 0f, 00, 1a, 00, 25, 00, 32, 18, 3d, 84, 00, 04, 41, 5b, 88, 00, 70, 2d, 18, 3d, 84, 00, 04, 43, 5b, 88, 00, 70, 22, 1a, Le: 00, SW1: 90, SW2: 00
CIA: 80, INS: b4, P1: 07, P2: 00, Lc: 20, 03, ae, 00, 38, 19, 03, 04, 8b, 00, 08, 70, 15, 19, 8b, 00, 09, 3b, 18, 1a, 08, 25, 88, 00, 70, 08, 11, 6d, 00, 8d, 00, 07, 7a, Le: 00, SW1: 90, SW2: 00
CIA: 80, INS: bc, P1: 07, P2: 00, Lc: 00, Le: 00, SW1: 90, SW2: 00
CIA: 80, INS: b2, P1: 08, P2: 00, Lc: 00, Le: 00, SW1: 90, SW2: 00
CIA: 80, INS: b4, P1: 08, P2: 00, Lc: 0d, 08, 00, 0a, 00, 00, 00, 00, 00, 00, 00, 00, 00, Le: 00, SW1: 90, SW2: 00
CIA: 80, INS: bc, P1: 08, P2: 00, Lc: 00, Le: 00, SW1: 90, SW2: 00
CIA: 80, INS: b2, P1: 05, P2: 00, Lc: 00, Le: 00, SW1: 90, SW2: 00
CIA: 80, INS: b4, P1: 05, P2: 00, Lc: 20, 05, 00, 2a, 00, 0a, 02, 00, 00, 00, 06, 80, 03, 00, 01, 00, 00, 00, 06, 00, 00, 01, 03, 80, 03, 01, 03, 80, 0a, 01, 03, 80, 03, Le: 00, SW1: 90, SW2: 00
CIA: 80, INS: b4, P1: 05, P2: 00, Lc: 0d, 03, 06, 80, 07, 01, 03, 80, 0a, 08, 03, 80, 0a, 06, Le: 00, SW1: 90, SW2: 00
CIA: 80, INS: bc, P1: 05, P2: 00, Lc: 00, Le: 00, SW1: 90, SW2: 00
CIA: 80, INS: b2, P1: 09, P2: 00, Lc: 00, Le: 00, SW1: 90, SW2: 00
CIA: 80, INS: b4, P1: 09, P2: 00, Lc: 18, 09, 00, 15, 00, 07, 0a, 3f, 05, 06, 05, 06, 14, 00, 0a, 05, 0a, 04, 03, 07, 05, 10, 33, 06, 0f, Le: 00, SW1: 90, SW2: 00
CIA: 80, INS: bc, P1: 09, P2: 00, Lc: 00, Le: 00, SW1: 90, SW2: 00
CIA: 80, INS: ba, P1: 00, P2: 00, Lc: 00, Le: 00, SW1: 90, SW2: 00

```

## Relançons CREF

```

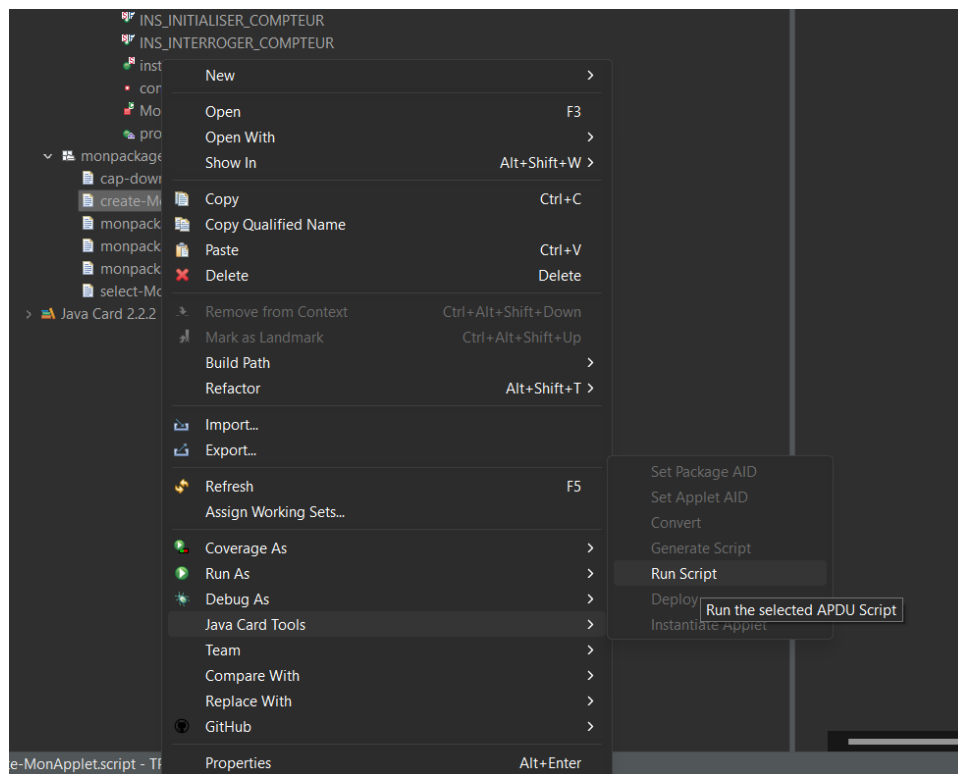
C:\Users\oussa>cref -i monapplet.eeprom
Java Card 2.2.2 C Reference Implementation Simulator (version 0.41)
32-bit Address Space implementation - with cryptography support
T=1 / T=CL Dual interface APDU protocol (ISO 7816-3)
Copyright 2005 Sun Microsystems, Inc. All rights reserved.

Memory configuration
  Type      Base      Size      Max Addr
  RAM       0x0       0x1000   0xffff
  ROM       0x2000   0xe000   0xffff
  E2P       0x10020   0xffe0   0x1ffff

  ROM Mask size =          0xce64 =          52836 bytes
  Highest ROM address in mask = 0xee63 =          61027 bytes
  Space available in ROM =    0x119c =          4508 bytes
EEPROM (0xffe0 bytes) restored from file "monapplet.eeprom"
Using a pre-initialized Mask
|

```

## Installons notre applet



```

<terminated> C:\Program Files\Java\jdk-21\bin\javaw.exe (24 oct. 2023, 11:13:18 AM) [pid: 8232]
Java Card 2.2.2 APDU Tool, Version 1.3
Copyright 2005 Sun Microsystems, Inc. All rights reserved. Use is subject to license terms.
Opening connection to localhost on port 9025.
Connected.
Received ATR = 0x3b 0xf0 0x11 0x00 0xff 0x01
CLA: 00, INS: a4, P1: 04, P2: 00, Lc: 09, a0: 00, 00, 00, 62, 03, 01, 08, 01, Le: 00, SW1: 90, SW2: 00
CLA: 80, INS: b8, P1: 00, P2: 00, Lc: 0d, 0b, 01, 02, 03, 04, 05, 06, 07, 08, 09, 00, 00, 00, Le: 0b, 01, 02, 03, 04, 05, 06, 07, 08, 09, 00, 00, SW1: 90, SW2: 00

```

relancer CREF (qui s'est terminé sur une commande powerdown)



```

C:\Users\oussa>cref -i monapplet.eeprom
Java Card 2.2.2 C Reference Implementation Simulator (version 0.41)
32-bit Address Space implementation - with cryptography support
T=1 / T=CL Dual interface APDU protocol (ISO 7816-3)
Copyright 2005 Sun Microsystems, Inc. All rights reserved.

Memory configuration
  Type      Base      Size      Max Addr
  RAM       0x0       0x1000    0xffff
  ROM       0x2000    0xe000    0xfffff
  E2P       0x10020   0xffe0    0x1ffff

  ROM Mask size =          0xce64 =          52836 bytes
  Highest ROM address in mask = 0xee63 =          61027 bytes
  Space available in ROM =    0x119c =           4508 bytes
EEPROM (0xffe0 bytes) restored from file "monapplet.eeprom"
Using a pre-initialized Mask

```

lançons apdutool, sélectionnons notre applet, après quoi nous pouvons envoyer des APDU à notre applet :

```

PS C:\> apdutool
Java Card 2.2.2 APDU Tool, Version 1.3
Copyright 2005 Sun Microsystems, Inc. All rights reserved. Use is subject to license terms.
Opening connection to localhost on port 9025.
Connected.
powerup;
// select MonApplet applet
Received ATR = 0x3b 0xf0 0x11 0x00 0xff 0x01
0x00 0xA4 0x04 0x00 0xb 0x01 0x02 0x03 0x04 0x05 0x06 0x07 0x08 0x09 0x00 0x00 0x7F;
CLA: 00, INS: a4, P1: 04, P2: 00, Lc: 0b, 01, 02, 03, 04, 05, 06, 07, 08, 09, 00, 00, Le: 00, SW1
: 90, SW2: 00

```

# Programmation d'une application coté client

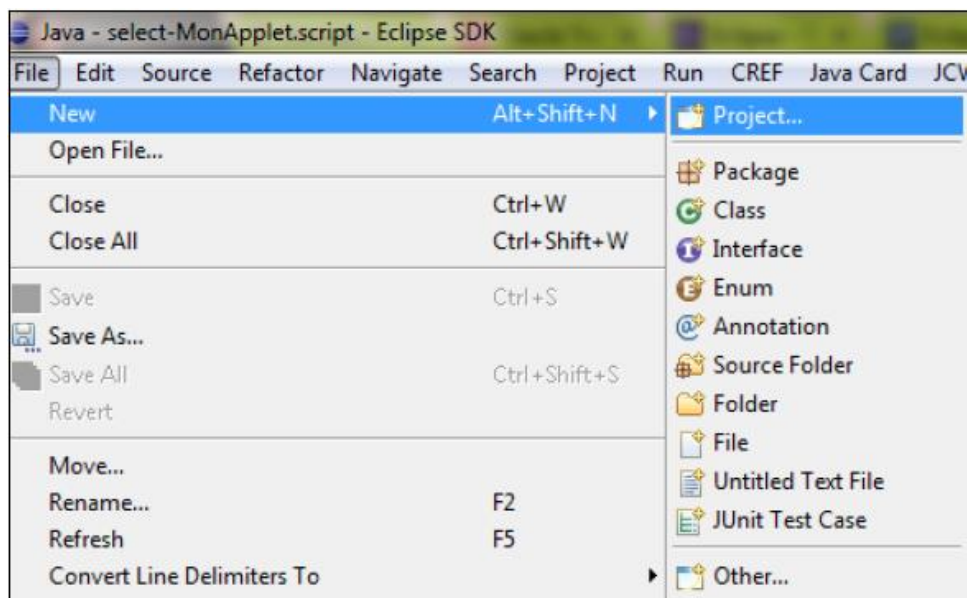
---

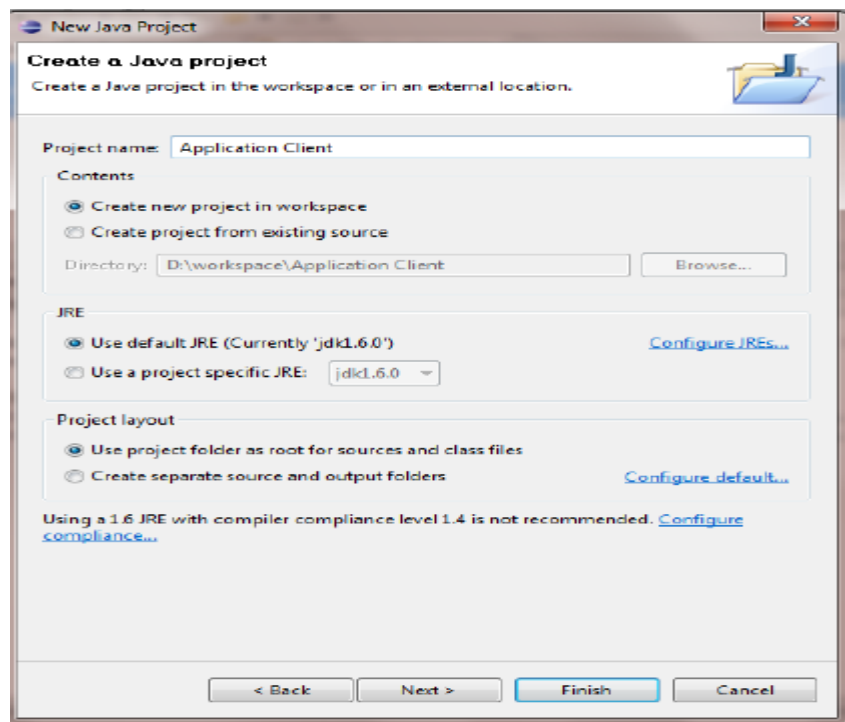
## **But de ce TP :**

Maintenant que nous avons programmé notre applet Javcard compteur, nous pouvons coder une application cliente : l'équivalent du terminal bancaire si notre Javacard était une carte de paiement.

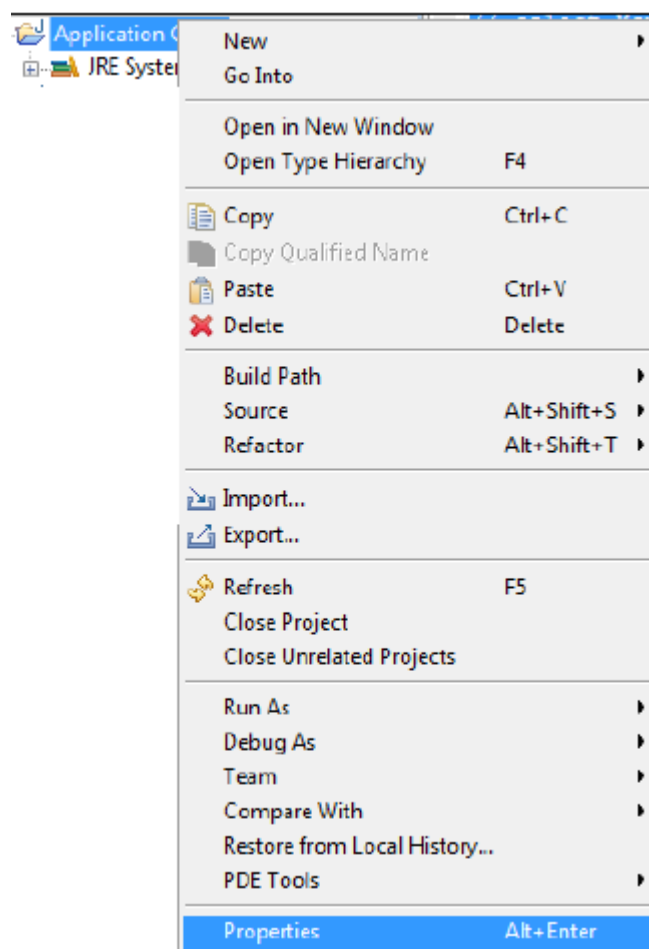
### I.1 Création de l'application client sous Eclipse :

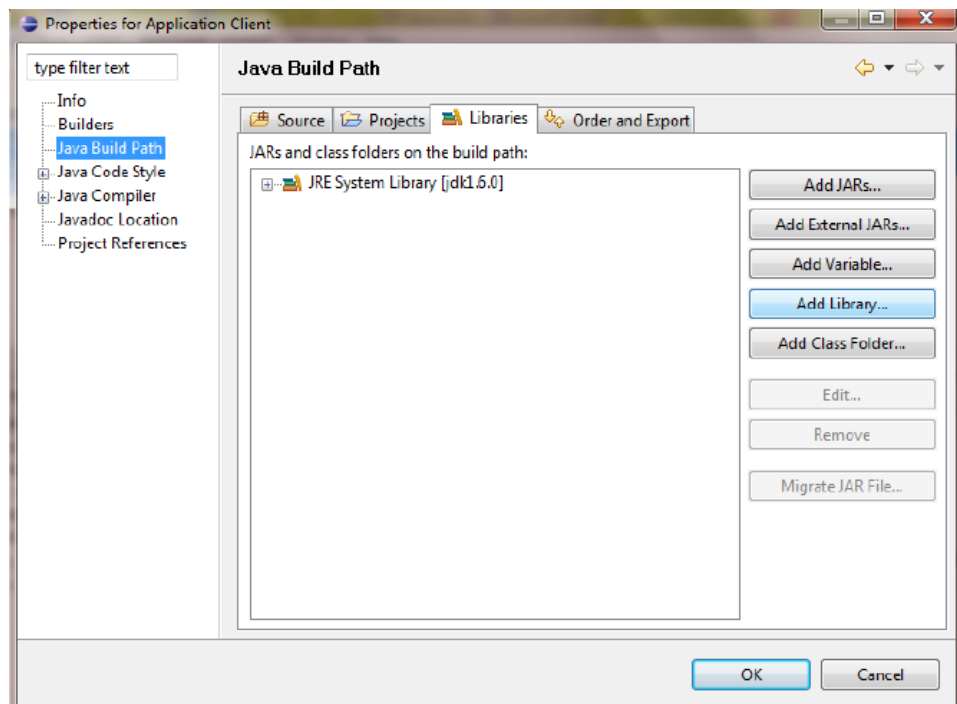
#### a) Création d'un nouveau projet :



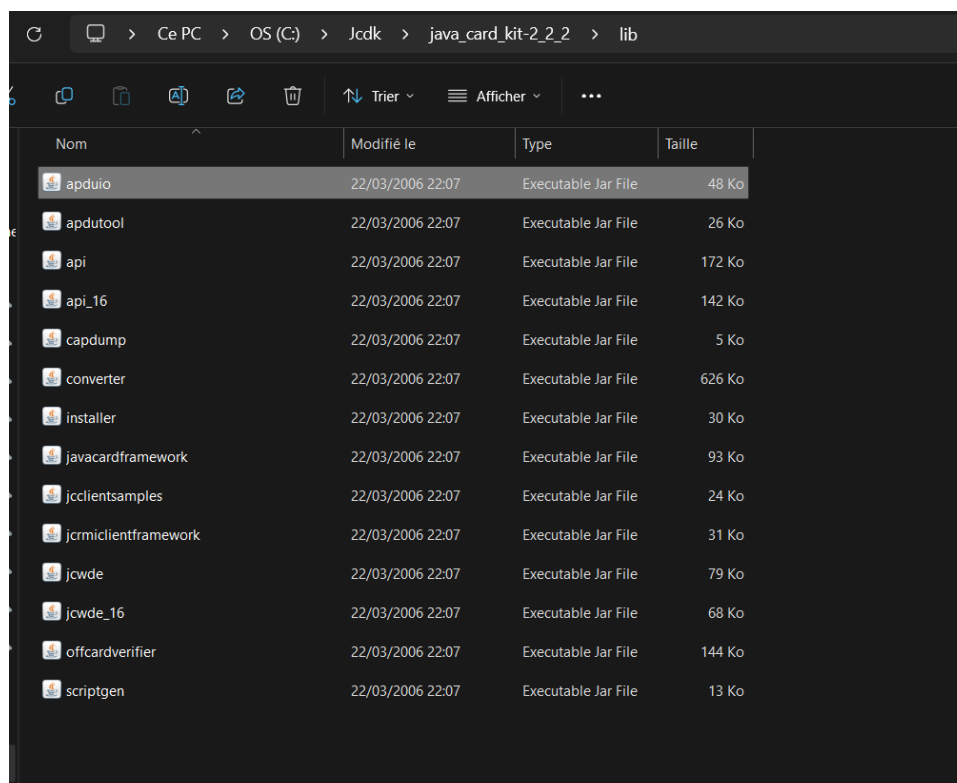


b) Ajout de la librairie « apduio » dans le classpath:

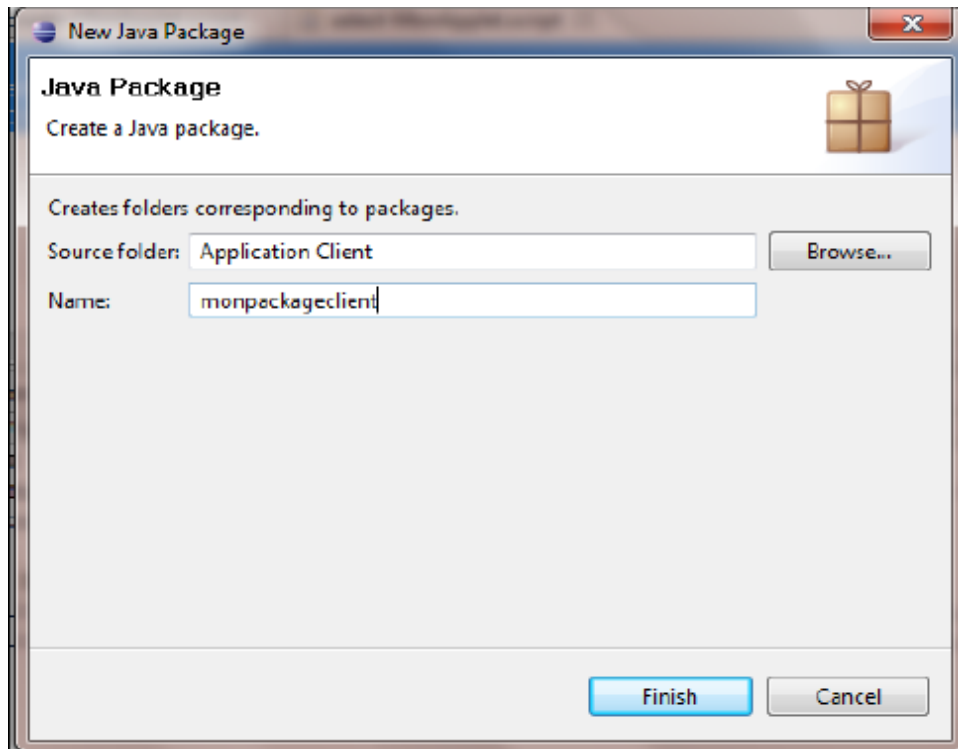
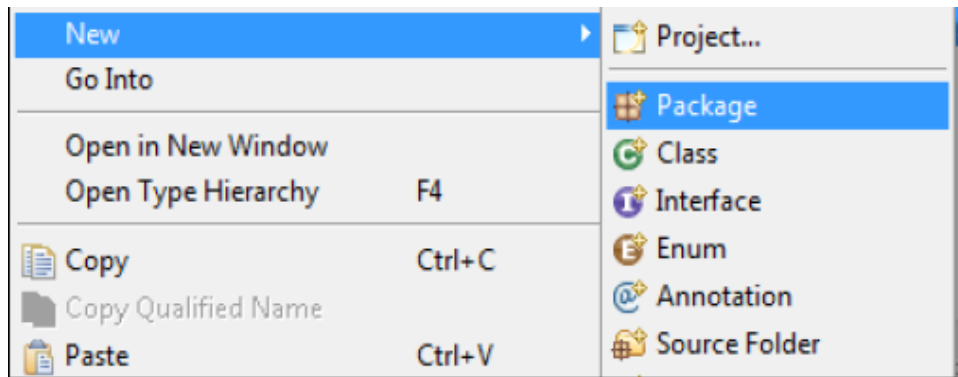




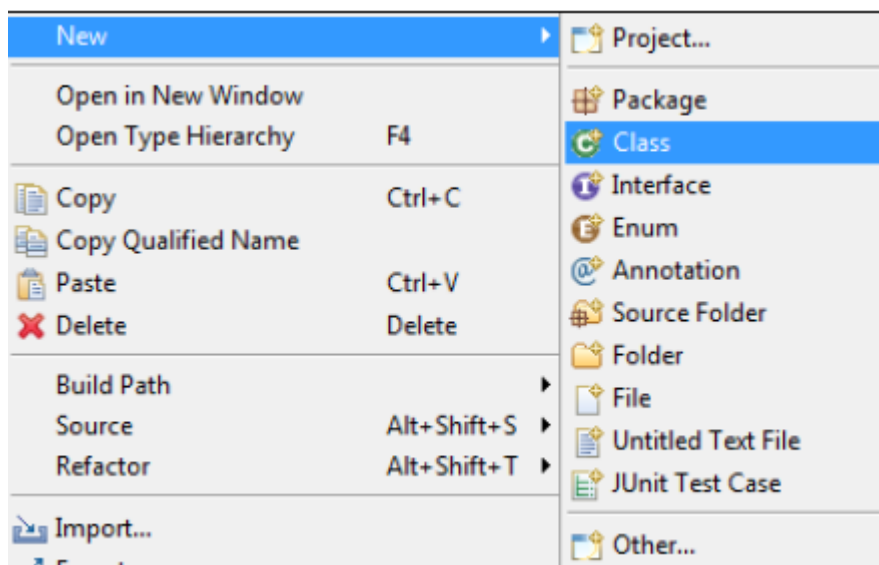
Sélectionner alors le fichier apduio.jar, valider et appuyer sur le bouton OK.

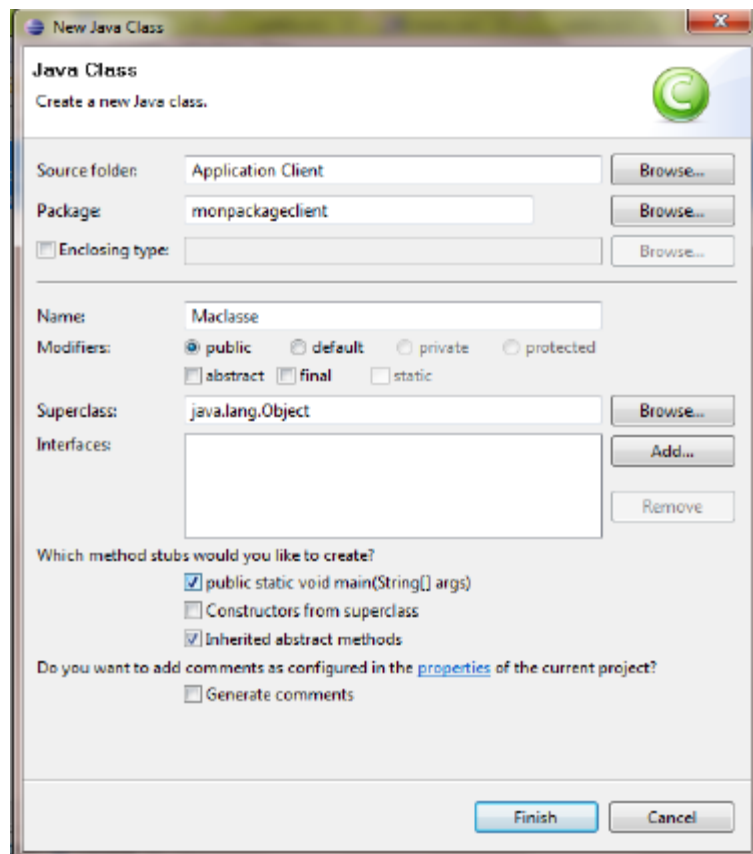


c) Création de la classe principale :



Créons maintenant la classe principale de notre application.





## Étape 1 - Connexion :

```

1 package monpackageclient;
2
3
4 import java.io.*;
5 import java.net.Socket;
6 import com.sun.javacard.apduio.Apdu;
7 import com.sun.javacard.apduio.CadTlClient;
8 import com.sun.javacard.apduio.CadTransportException;
9
10 public class Maclasse {
11     public static final byte CLA_MONAPPLET = (byte) 0xB0;
12     public static final byte INS_INCREMENTER_COMPTEUR = 0x00;
13     public static final byte INS_DECREMENTER_COMPTEUR = 0x01;
14     public static final byte INS_INTERROGER_COMPTEUR = 0x02;
15     public static final byte INS_INITIALISER_COMPTEUR = 0x03;
16
17     public static void main(String[] args) throws IOException, CadTransportException {
18         /* Connexion - Javacard */
19         CadTlClient cad;
20         Socket sckCarte;
21         try {
22             sckCarte = new Socket("localhost", 9025);
23             sckCarte.setTcpNoDelay(true);
24             BufferedInputStream input = new BufferedInputStream(sckCarte.getInputStream());
25             BufferedOutputStream output = new BufferedOutputStream(sckCarte.getOutputStream());
26             cad = new CadTlClient(input, output);
27         } catch (Exception e) {
28             System.out.println("Erreur : impossible de se connecter a la Javacard");
29             return;
30         }
31         /* Mise sous tension de la carte */
32         try {
33             cad.powerUp();
34         } catch (Exception e) {
35             System.out.println("Erreur lors de l'envoi de la commande Powerup a la Javacard");
36             return;
37         }
38     }

```

## Etape 2 - Sélection:

```

39      /* Sélection de l'applet */
40      Apdu apdu = new Apdu();
41      apdu.command[Apdu.CLA] = 0x00;
42      apdu.command[Apdu.INS] = (byte) 0xA4;
43      apdu.command[Apdu.P1] = 0x04;
44      apdu.command[Apdu.P2] = 0x00;
45      byte[] appletAID = { 0x01, 0x02, 0x03, 0x04, 0x05, 0x06,
46                          0x07, 0x08, 0x09, 0x00, 0x00 };
47      apdu.setDataIn(appletAID);
48      cad.exchangeApdu(apdu);
49      if (apdu.getStatus() != 0x9000) {
50          System.out.println("Erreur lors de la sélection de l'applet");
51          System.exit(1);
52      }
53

```

### Etape 3 - Invocation des services implémentés:

```

index.html  MonApplet.java  module-info.java  Maclasse.java
54      /* Menu principal */
55      boolean fin = false;
56      while (!fin) {
57          System.out.println();
58          System.out.println("Application cliente Javacard");
59          System.out.println("-----");
60          System.out.println();
61          System.out.println("1 - Interroger le compteur");
62          System.out.println("2 - Incrémenter le compteur");
63          System.out.println("3 - Décrémenter le compteur");
64          System.out.println("4 - Réinitialiser le compteur");
65          System.out.println("5 - Quitter");
66          System.out.println();
67          System.out.println("Votre choix ?");
68          int choix = System.in.read();
69          while (!(choix >= '1' && choix <= '5')) {
70              choix = System.in.read();
71          }
72          apdu = new Apdu();
73          apdu.command[Apdu.CLA] = Maclasse.CLA_MONAPPLET;
74          apdu.command[Apdu.P1] = 0x00;
75          apdu.command[Apdu.P2] = 0x00;
76          apdu.setLe(0x7F);
77          switch (choix) {
78              case '1':
79                  apdu.command[Apdu.INS] =
80                      Maclasse.INS_INTERROGER_COMPTEUR;
81                  cad.exchangeApdu(apdu);
82                  if (apdu.getStatus() != 0x9000) {
83                      System.out.println("Erreur : status word different de 0x9000");
84                  } else {
85                      System.out.println("Valeur du compteur : " + apdu.dataOut[0]);
86                  }
87                  break;
88              case '2':
89                  apdu.command[Apdu.INS] = Maclasse.INS_INCREMENTER_COMPTEUR;
90                  cad.exchangeApdu(apdu);
91                  if (apdu.getStatus() != 0x9000) {
92                      System.out.println("Erreur : status word different de 0x9000");
93                  } else {
94                      System.out.println("OK");
95                  }
96                  break;
97              case '3':
98                  apdu.command[Apdu.INS] = Maclasse.INS_DECREMENTER_COMPTEUR;
99                  cad.exchangeApdu(apdu);

```

```

          break;
      case '3':
          apdu.command[Apdu.INS] = Maclasse.INS_DECREMENTER_COMPTEUR;
          cad.exchangeApdu(apdu);
          if (apdu.getStatus() != 0x9000) {
              System.out.println("Erreur : status word different de 0x9000");
          } else {
              System.out.println("OK");
          }
          break;
      case '4':
          apdu.command[Apdu.INS] = Maclasse.INS_INITIALISER_COMPTEUR;
          byte[] donnees = new byte[1];
          donnees[0] = 0;
          apdu.setDataIn(donnees);
          cad.exchangeApdu(apdu);
          if (apdu.getStatus() != 0x9000) {
              System.out.println("Erreur : status word different de 0x9000");
          } else {
              System.out.println("OK");
          }
          break;
      case '5':
          fin = true;
          break;
      }
  }
}

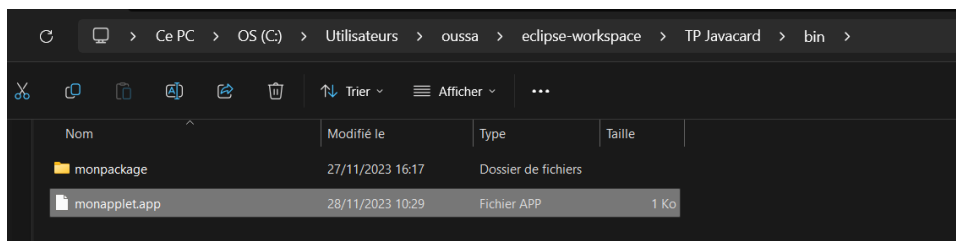
```

#### Etape 4 - Mise hors tension:

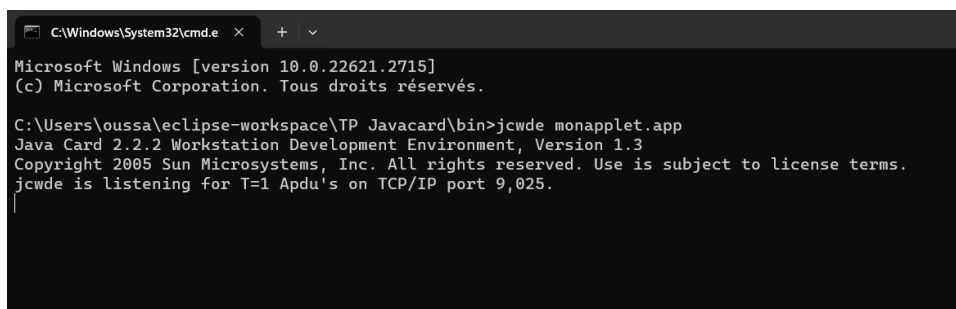
```
123
124      /* Mise hors tension de la carte */
125      try {
126          cad.powerDown();
127      } catch (Exception e) {
128          System.out.println("Erreur lors de l'envoi de la commande Powerdown a la Javacard");
129          return;
130      }
131
132
133
134    }
135
136 }
```

#### I.2 Utilisation de l'application cliente avec un simulateur – JCWDE:

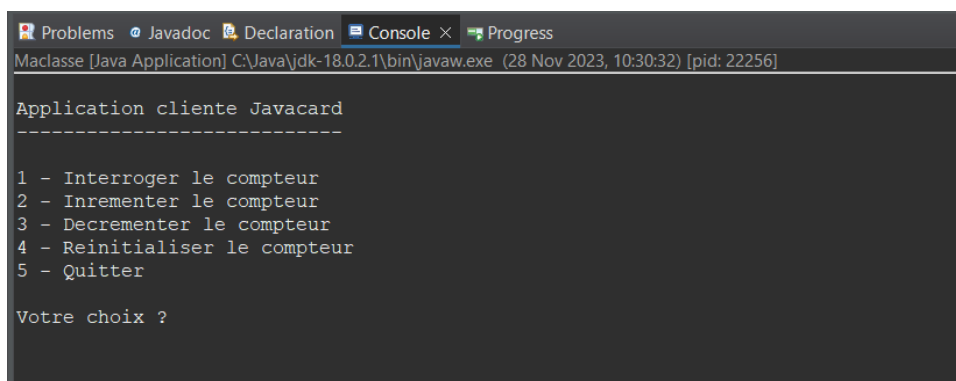
##### créer un fichier "de configuration:



##### lancer notre simulateur:



##### lançons notre application cliente :



##### interroger le compteur :



```
Problems Javadoc Declaration Console × Progress
Maclasse [Java Application] C:\Java\jdk-18.0.2.1\bin\javaw.exe (28 Nov 2023, 10:30:32) [pid: 22256]

Application cliente Javacard
-----

1 - Interroger le compteur
2 - Inrementer le compteur
3 - Decrementer le compteur
4 - Reinitialiser le compteur
5 - Quitter

Votre choix ?
1
Valeur du compteur : 0

Application cliente Javacard
-----

1 - Interroger le compteur
2 - Inrementer le compteur
3 - Decrementer le compteur
4 - Reinitialiser le compteur
5 - Quitter

Votre choix ?
```

Incrémentons maintenant le compteur 3 fois puis décrémentons-le une fois:

```
Problems Javadoc Declaration Console × Progress
Maclasse [Java Application] C:\Java\jdk-18.0.2.1\bin\javaw.exe (28 Nov 2023, 10:30:32) [pid: 22256]

Votre choix ?
3
OK

Application cliente Javacard
-----

1 - Interroger le compteur
2 - Inrementer le compteur
3 - Decrementer le compteur
4 - Reinitialiser le compteur
5 - Quitter

Votre choix ?
1
Valeur du compteur : 2

Application cliente Javacard
-----

1 - Interroger le compteur
2 - Inrementer le compteur
3 - Decrementer le compteur
4 - Reinitialiser le compteur
5 - Quitter

Votre choix ?
```

Quittons maintenant notre application cliente:

```
C:\Windows\System32\cmd.e x + v
Microsoft Windows [version 10.0.22621.2715]
(c) Microsoft Corporation. Tous droits réservés.

C:\Users\oussa\eclipse-workspace\TP Javacard\bin>jcwde monapplet.app
Java Card 2.2.2 Workstation Development Environment, Version 1.3
Copyright 2005 Sun Microsystems, Inc. All rights reserved. Use is subject to license terms.
jcwde is listening for T=1 Adu's on TCP/IP port 9,025.
jcwde exiting on receipt of power down command.

C:\Users\oussa\eclipse-workspace\TP Javacard\bin>|
```

## **Mini projet java card**

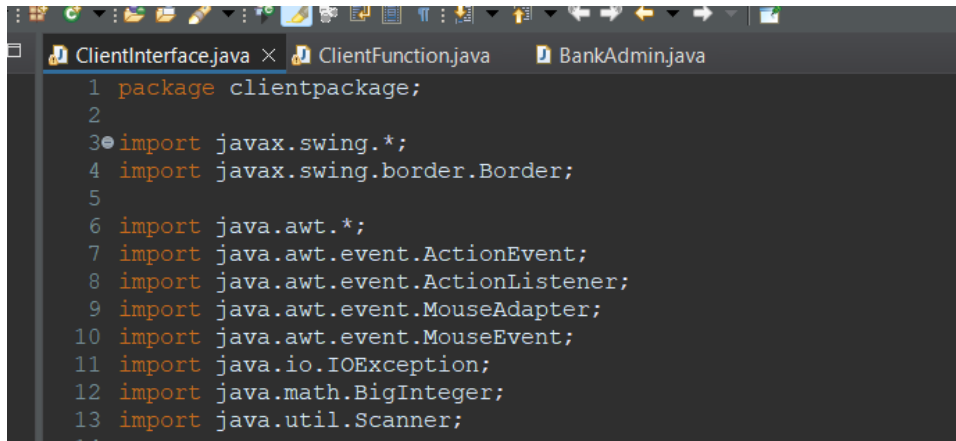
---

### I- Partie client:

#### I-1- préparation de l'interface graphique:

On a utiliser la bibliothèque de java, SWING pour construire l'interface graphique.

On a utiliser aussi autres bibliothèque pour ajouter les évènements nécessaires a notre interface graphique.

A screenshot of an IDE window showing the imports for ClientInterface.java. The imports are: package clientpackage; import javax.swing.\*; import javax.swing.border.Border; import java.awt.\*; import java.awt.event.ActionEvent; import java.awt.event.ActionListener; import java.awt.event.MouseAdapter; import java.awt.event.MouseEvent; import java.io.IOException; import java.math.BigInteger; import java.util.Scanner;

```
1 package clientpackage;
2
3 import javax.swing.*;
4 import javax.swing.border.Border;
5
6 import java.awt.*;
7 import java.awt.event.ActionEvent;
8 import java.awt.event.ActionListener;
9 import java.awt.event.MouseAdapter;
10 import java.awt.event.MouseEvent;
11 import java.io.IOException;
12 import java.math.BigInteger;
13 import java.util.Scanner;
```

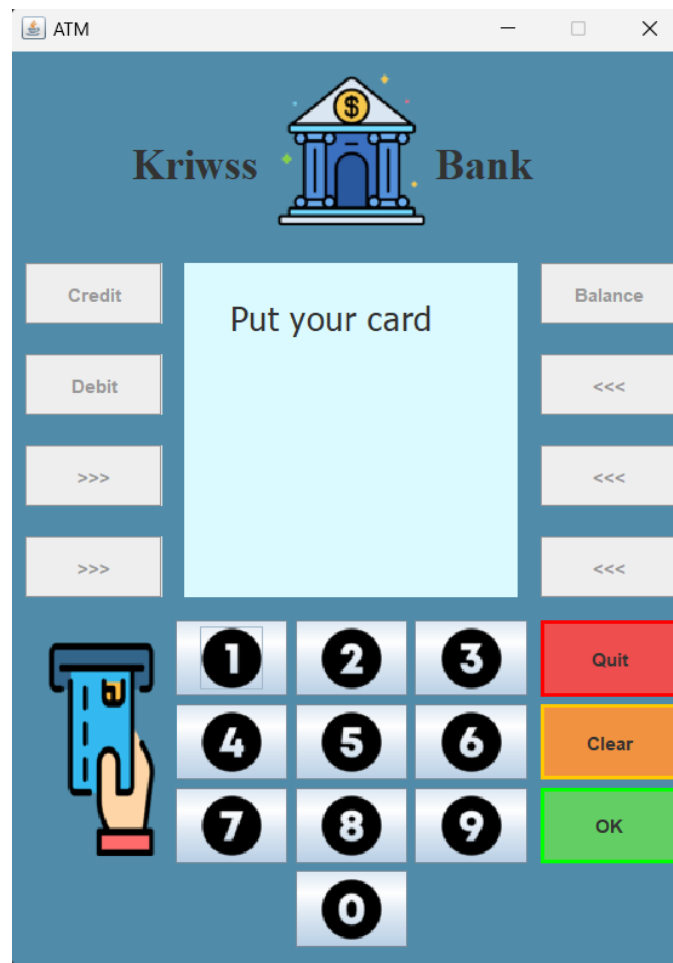
#### **1-explication de l'importations des différents bibliothèques:**

- “**import javax.swing.\*;**”: Importe l'ensemble du package **javax.swing**, qui fournit des classes pour construire des interfaces graphiques.
- “**import javax.swing.border.Border;**”: Importe la classe **Border** du package **javax.swing.border**, qui est utilisée pour définir des bordures sur les composants Swing.
- “**import java.awt.\*;**”: Importe l'ensemble du package **java.awt**, qui contient des classes pour créer des composants GUI de base et gérer les événements.

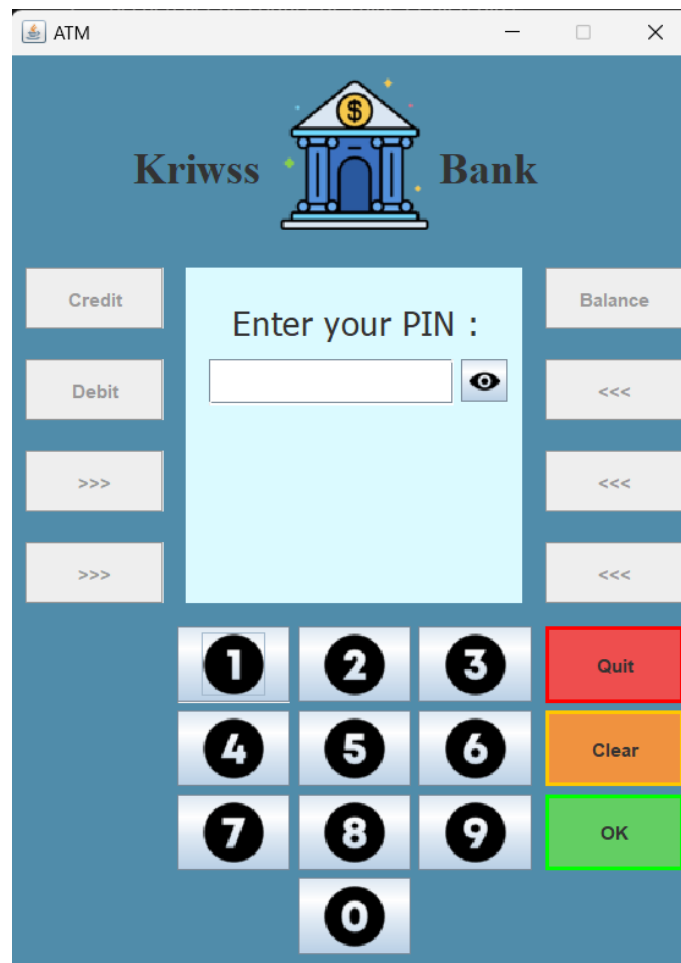
- **“import java.awt.event.ActionEvent;”**: Importe la classe **ActionEvent** du package **java.awt.event**, qui est utilisée pour gérer les actions (par exemple, les clics sur un bouton).
- **“import java.awt.event.ActionListener;”**: Importe l'interface **ActionListener** du package **java.awt.event**, qui est utilisée pour gérer les événements d'action.
- **“import java.awt.event.MouseAdapter;”**: Importe la classe **MouseAdapter** du package **java.awt.event**, qui est une classe utilitaire pour recevoir des événements de la souris.
- **“import java.awt.event.MouseEvent;”**: Importe la classe **MouseEvent** du package **java.awt.event**, qui représente des événements de la souris.
- **“import java.io.IOException;”**: Importe la classe **IOException**, qui est utilisée pour gérer les exceptions d'entrée/sortie.
- **“import java.math.BigInteger;”**: Importe la classe **BigInteger**, qui fournit des opérations pour manipuler de grands entiers.
- **“import java.util.Scanner;”**: Importe la classe **Scanner**, qui est utilisée pour lire l'entrée de l'utilisateur.

## I-2- Les différents interfaces de notre application:

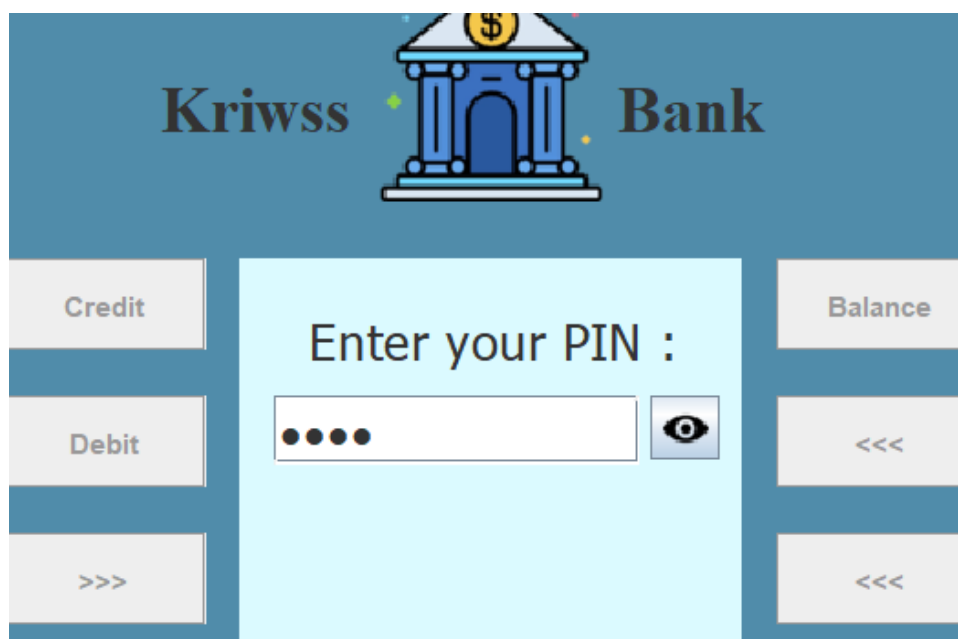
La première interface qui s’affiche lors de l’exécution. Il faut insérer la carte.



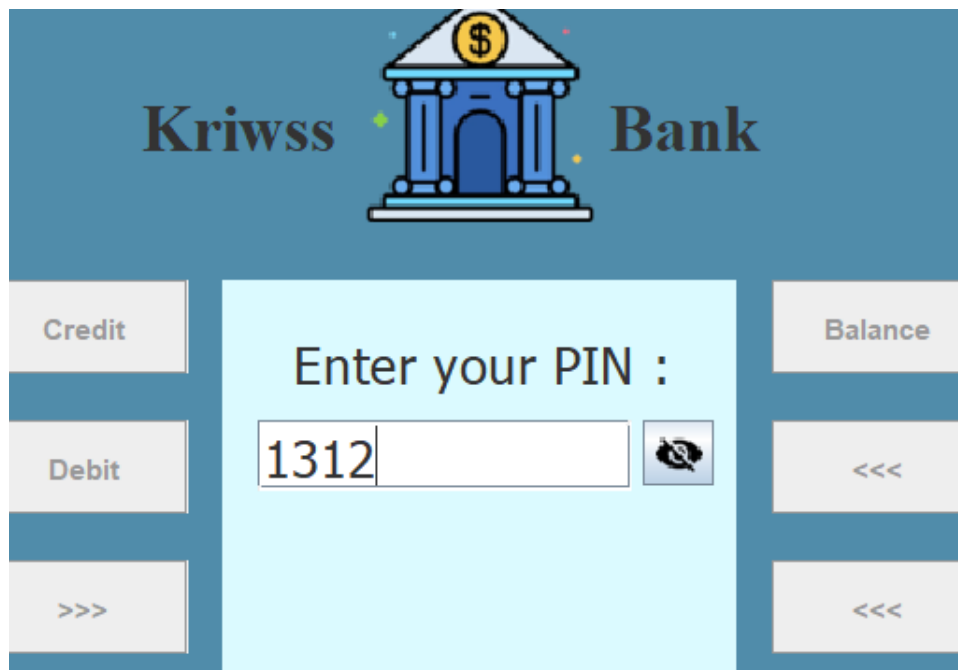
En insérant la carte, on va avoir une 2 -ème interface. Il faut saisir le code pin dans la zone de texte puis valider ( en cliquant sur le bouton ok ). un apdu sera envoyer au serveur pour confirmer si le code pin est correct ou non.



On insère le code pin. En cliquant sur le bouton de l'oeil on peut voir le code tapée ou rendre invisible.

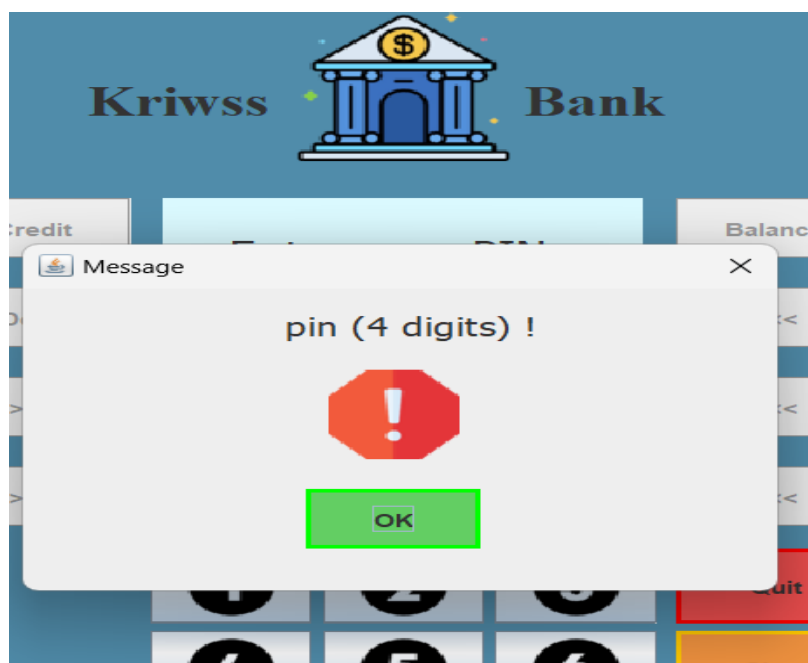


Où:

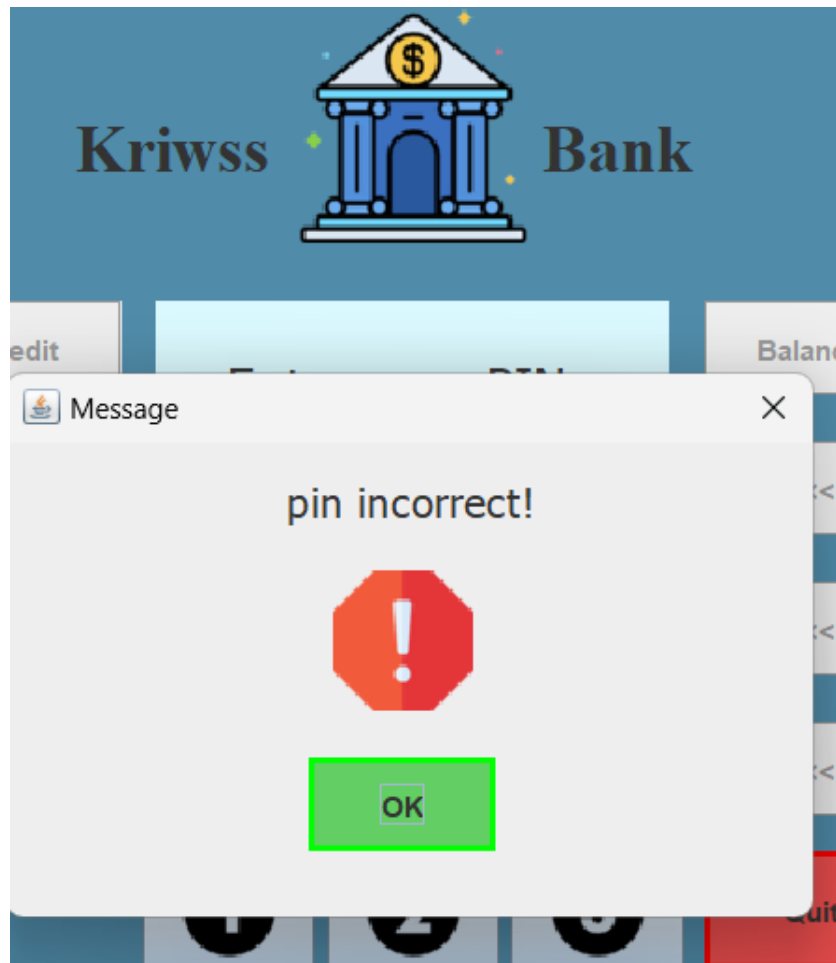


On peut avoir plusieurs cas.

1 ère cas: si la longueur de code pin est inferieure à 4. une fenêtre d'erreur s'affiche.

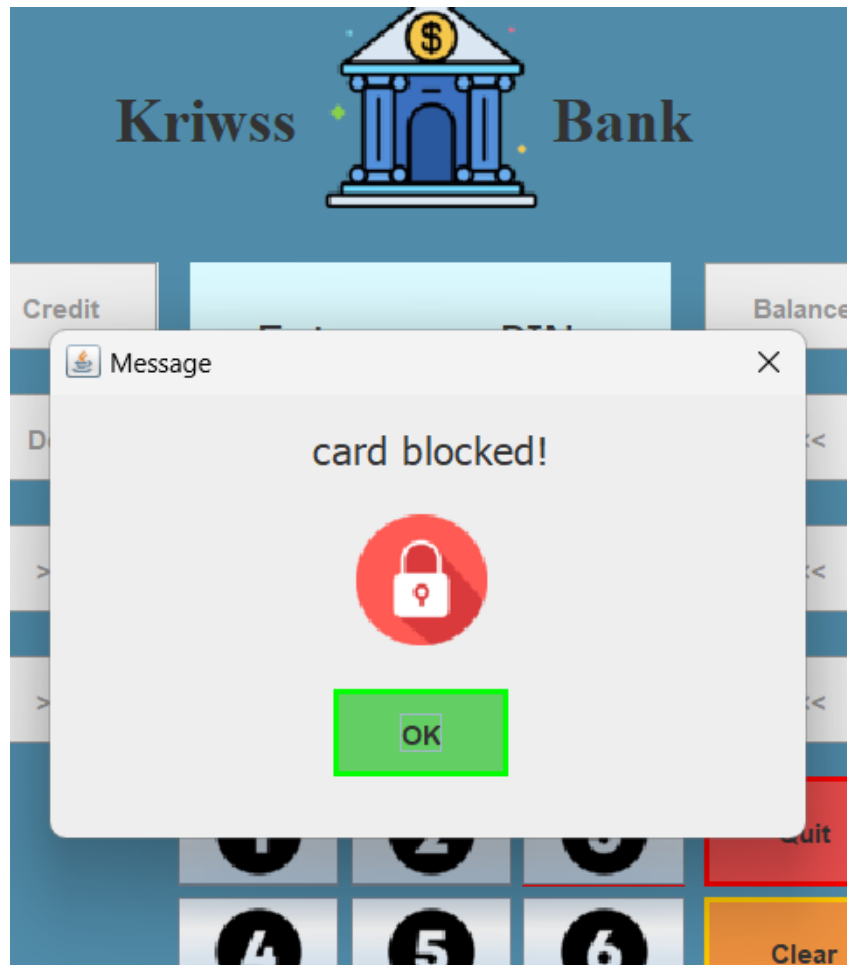


2 -ème cas: le code pin est de longueur 4 mais il est incorrecte. Une fenêtre d'erreur va s'afficher.



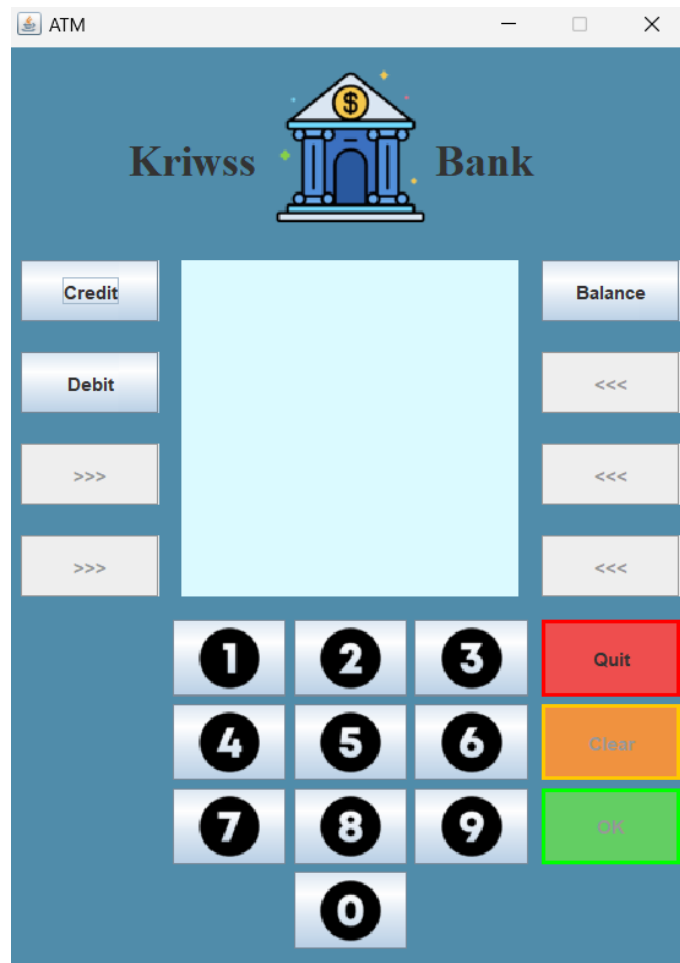
3 -ème cas: si on répète 3 fois incorrectement le code pin, la carte va être bloquée et une fenêtre d'erreur va être afficher.





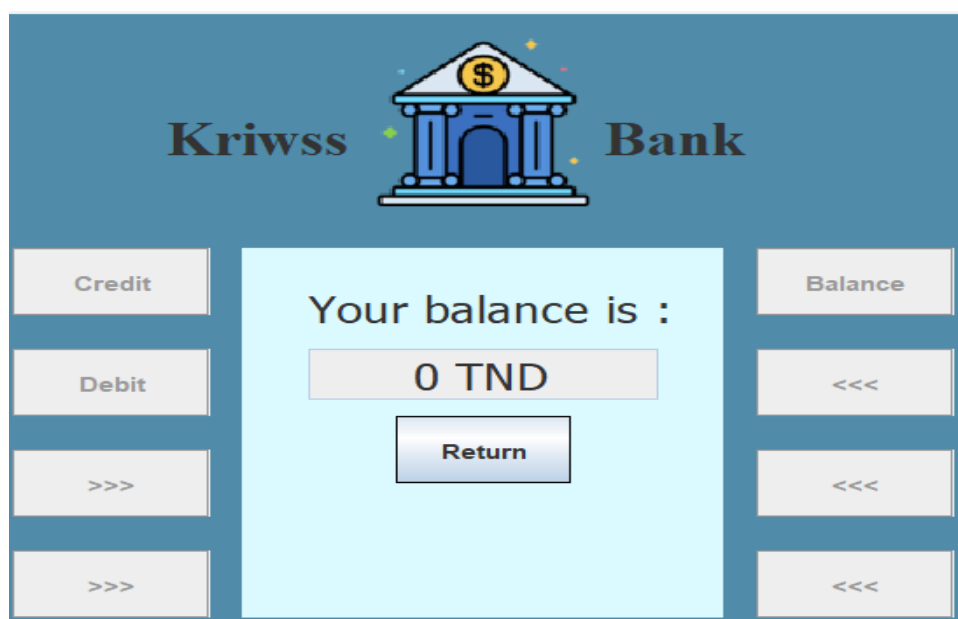
(Il faut quitter et redémarrer de nouveau.)

4 -ème cas: si le code pin est correcte, une 3 -ème interface va être afficher à l'utilisateur.



Dans cette interface on a 3 bouton disponible pour l'utilisateur. Soit crédit qui permet de créditer, soit débiter, qui permet au utilisateur de débiter de l'argent, soit balance, qui permet au utilisateur de voir son solde.

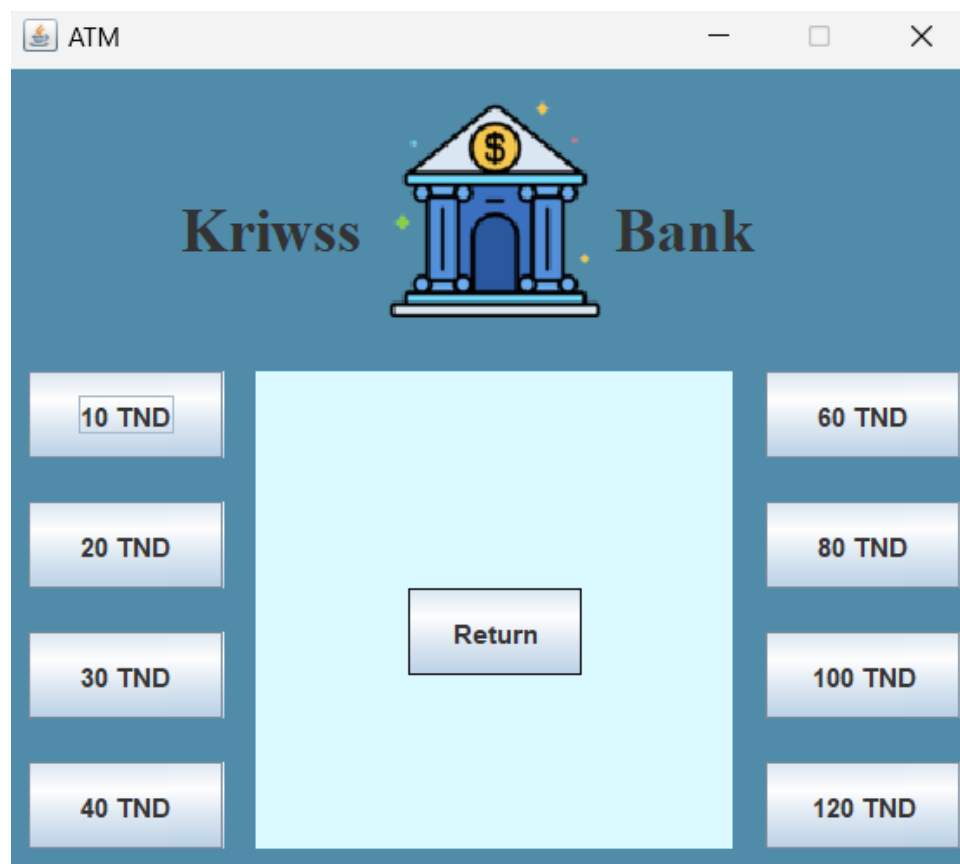
On cliquant sur le bouton balance une nouvelle interface va être afficher.



On clique sur le bouton return pour retourner à la page d'accueil.

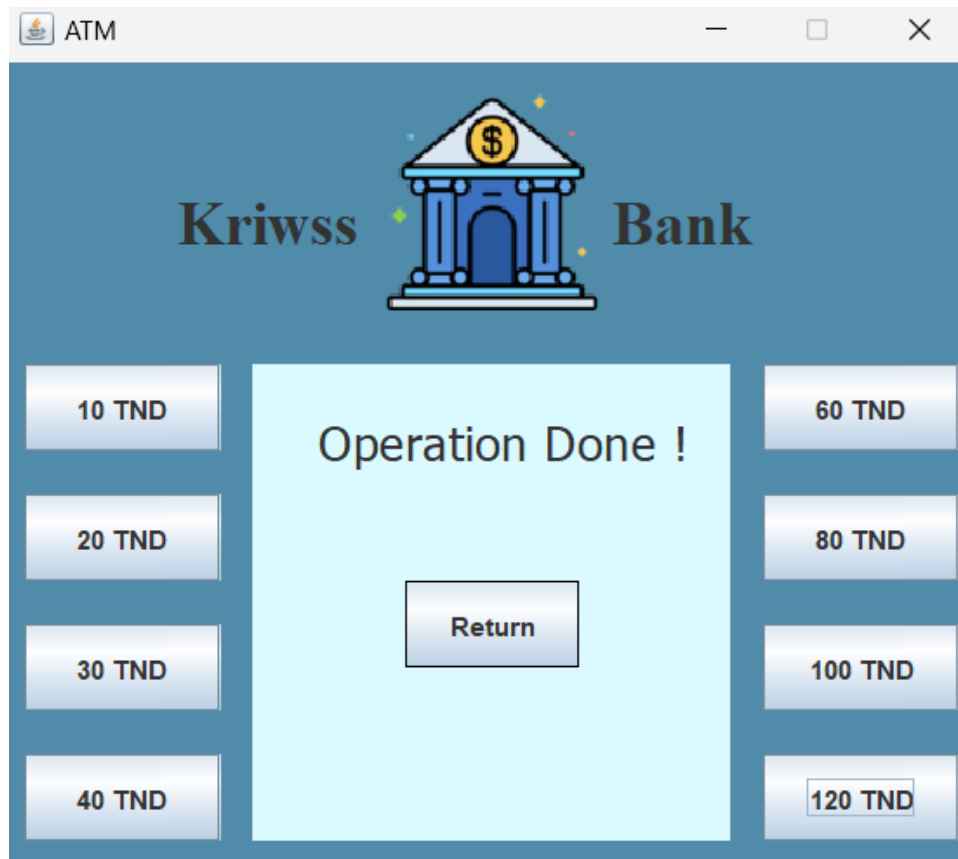
On clique maintenant sur le 2 -ème bouton, crédit. Qui permet de créditer de l'argent.

Une nouvelle interface est affichée maintenant.



On choisit la somme qu'on veut créditer et on tape sur elle.

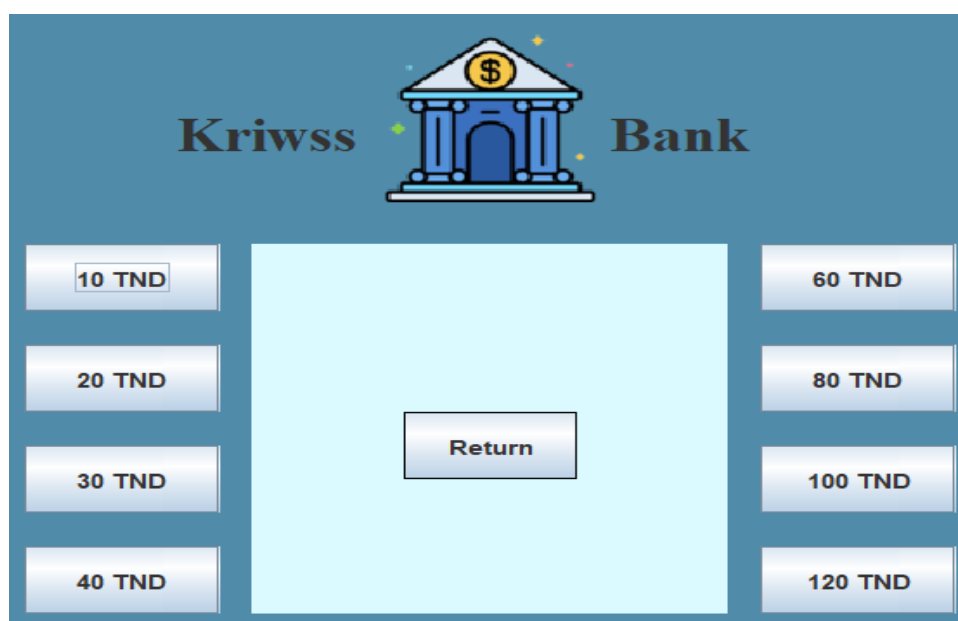
Un message va être afficher.



On clique sur le bouton return pour retourner à la page d'accueil.

On clique maintenant sur le 3 -ème bouton, débit. Qui permet de débiter de l'argent.

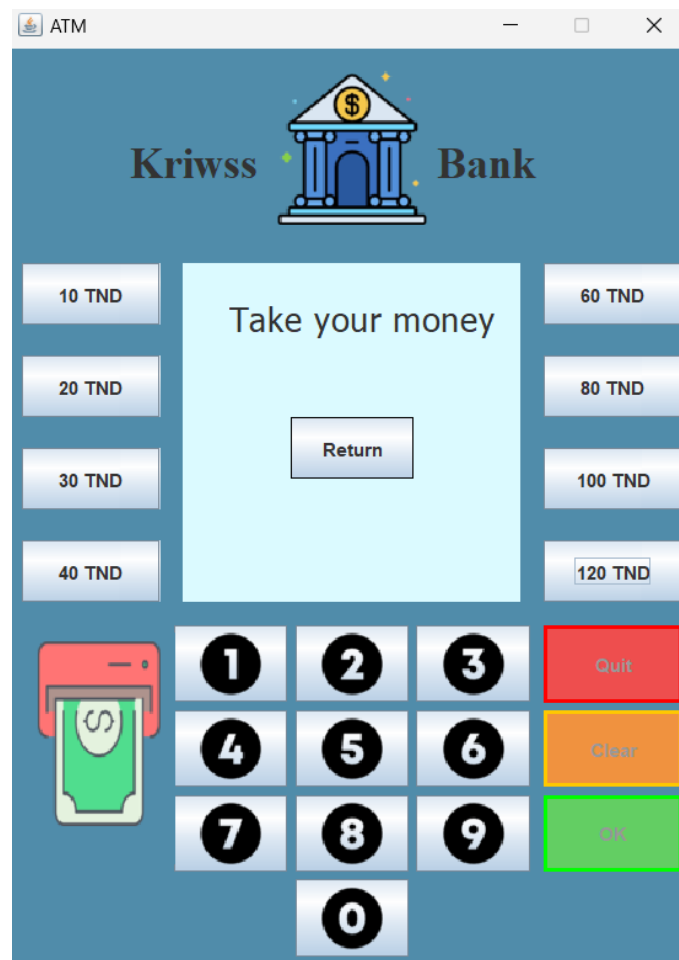
Une nouvelle interface est affichée maintenant.



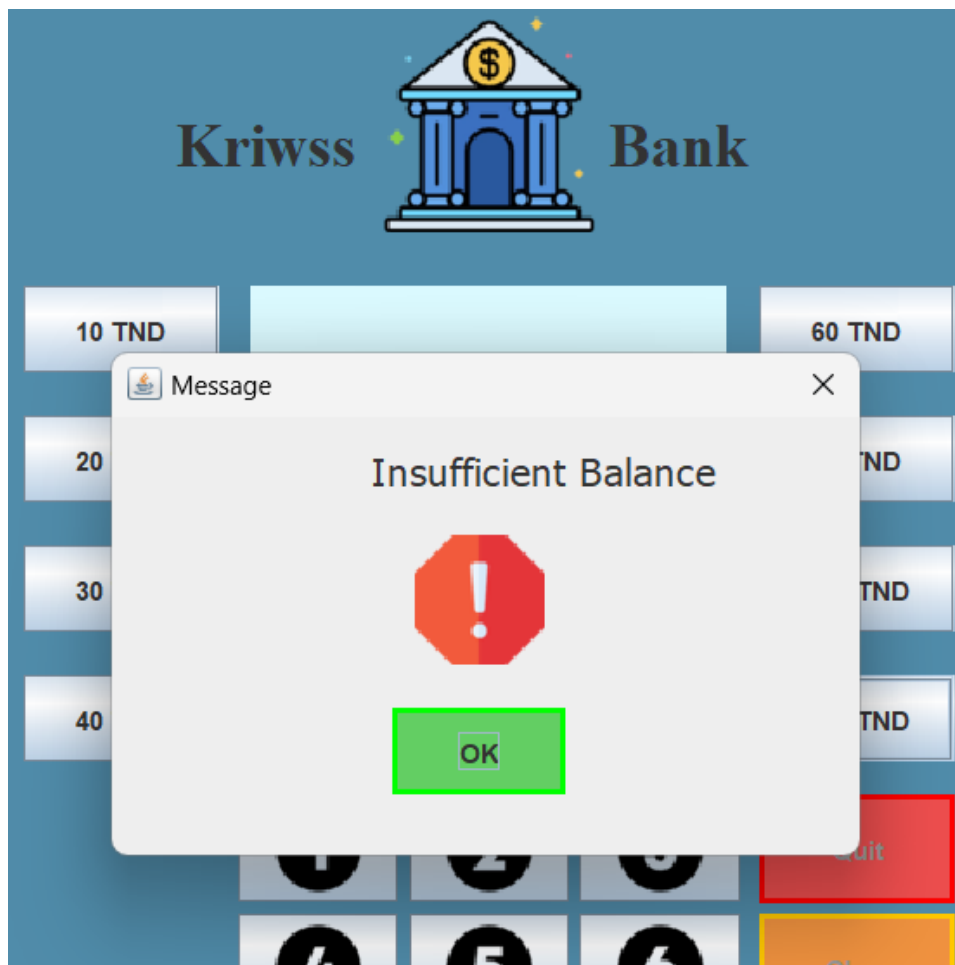
On choisit la somme qu'on veut débiter et on tape sur elle.

On a 2 cas possible, soit une cas de validation, soit une cas d'erreur.

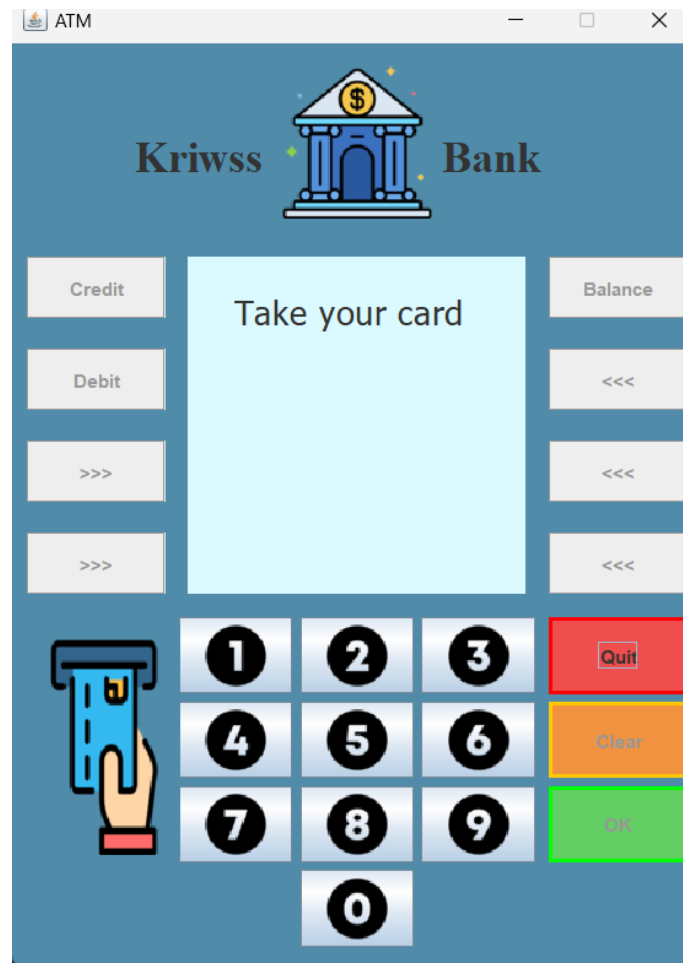
1 ère cas: une cas de validation. Le somme demander vas être à la position de client. Et il faut que le client prend la somme demandé en tapant sur l'icone d'argent.



2 -ème cas: cas d'erreur. Un message d'erreur va être afficher:



On clique maintenant sur le bouton quitter, un message va être affiché à l'utilisateur qui demande de lui de prendre sa carte.



On clique maintenant sur l'icone de la carte et on clique une autre fois sur le button quitter pour quitter.

```
C:\Users\oussa\eclipse-workspace\ATM-MasterCard\ATM-MasterCard\bin>jcwde monapplet.app
Java Card 2.2.2 Workstation Development Environment, Version 1.3
Copyright 2005 Sun Microsystems, Inc. All rights reserved. Use is subject to license terms.
jcwde is listening for T=1 Adu's on TCP/IP port 9,025.
jcwde exiting on receipt of power down command.
```

## I-3- implémentation des méthodes nécessaires:

Déclarations des variables qu'on va utiliser.

```
1 package clientpackage;
2
3 import java.io.BufferedReader;
4 import java.io.BufferedOutputStream;
5 import java.io.IOException;
6 import java.net.Socket;
7
8
9 import com.sun.javacard.apduio.Apdu;
10 import com.sun.javacard.apduio.CadTlClient;
11 import com.sun.javacard.apduio.CadTransportException;
12
13 public class ClientFunction {
14
15     static Apdu apdu ;
16     static CadTlClient cad;
17 }
```

Implémentation de la méthode msg():

Cette méthode permet l'envoi des APDU.

```
public Apdu Msg(byte ins, byte lc, byte[] data, byte le) throws IOException, CadTransportException{
    apdu = new Apdu();
    apdu.command[Apdu.CLA] = (byte) 0xB0;
    apdu.command[Apdu.P1] = 0x00;
    apdu.command[Apdu.P2] = 0x00;
    apdu.command[Apdu.P3] = 0x00;
    apdu.command[Apdu.INS] = ins;
    //apdu.setLe(0x7F);
    apdu.setLe(le);
    if (data!=null)
        apdu.setDataIn(data);
    cad.exchangeApdu(apdu);
    return apdu;
}
```

Implémentation de la méthode connect():

Cette méthode permet la connexion avec la carte.



```

public void Connect(){
    Socket sckCarte;

    try {
        sckCarte = new Socket("localhost", 8025);
        sckCarte.setTcpNoDelay(true);
        BufferedInputStream input = new BufferedInputStream(sckCarte.getInputStream());
        BufferedOutputStream output = new BufferedOutputStream(sckCarte.getOutputStream());
        cad = new CadTlClient(input, output);
    } catch (Exception e) {
        System.out.println("Erreur : impossible de se connecter a la Javacard");
        return;
    }

    /* Mise sous tension de la carte */
    try {
        cad.powerUp();
    } catch (Exception e) {
        System.out.println("Erreur lors de l'envoi de la commande Powerup a la Javacard");
        return;
    }
}

```

Implémentation de la méthode select():

Cette méthode permet la sélection de notre applet dans la carte.

```

public void select() throws IOException, CadTransportException{

    /* Sélection de l'applet :création du commande SELECT APDU */
    apdu = new Apdu();
    apdu.command[Apdu.CLA] = (byte) 0x00;
    apdu.command[Apdu.INS] = (byte) 0xA4;
    apdu.command[Apdu.P1] = 0x04;
    apdu.command[Apdu.P2] = 0x00;
    byte[] appletAID = { 0x01, 0x02, 0x03, 0x04, 0x05, 0x06, 0x07, 0x08, 0x09, 0x00, 0x00 };
    apdu.setDataIn(appletAID);
    cad.exchangeApdu(apdu);
    if (apdu.getStatus() != 0x9000) {
        System.out.println("Erreur lors de la sélection de l'applet");
        System.exit(1);
    }
}

```

Implémentation de la méthode deselect():

```

public void Deselect(){
    /* Mise hors tension de la carte */
    try {
        cad.powerDown();
    } catch (Exception e) {
        System.out.println("Erreur lors de l'envoi de la commande Powerdown a la Javacard");
        return;
    }
}

```

## II- Partie serveur:

### I-1- Déclaration des variables et des constantes:

```
10 public class BankAdmin extends Applet {
11
12     /* Constants */
13     public static final byte CLA_MONAPPLET = (byte) 0xB0;
14     public static final byte INS_TEST_CODE_PIN = 0x00;
15     public static final byte INS_INTERROGER_COMPTE = 0x01;
16     public static final byte INS_INCREMENTER_COMPTE = 0x02;
17     public static final byte INS_DECREMENTER_COMPTE = 0x03;
18     public static final byte INS_INITIALISER_COMPTE = 0x04;
19
20     public final static short MAX_BALANCE = 0x03E8; // le maximum de la balance (1000 TND)
21
22     public final static byte MAX_MONTANT_TRANSACTION = (byte) 127; // maximum montant
23                                     // qu'on peut
24                                     // transiter
25
26     public final static byte MAX_ERROR_PIN = (byte) 0x03; // maximum de code pin
27                                     // erroner
28
29     public final static byte MAX_PIN_LENGTH = (byte) 0x04; // longueur maximale du
30                                     // code pin
31
32
33     private byte[] INIT_PIN = { (byte) 1, (byte) 3, (byte) 1, (byte) 2 };
34
35     /* Exception */
36
37     // Verification Pin Echoué
38     final static short SW_VERIFICATION_FAILED = 0x6300;
39
40     final static short SW_EXCEED_TRY_LIMIT = 0x6321;
41     // signal the the PIN validation is required
42     // for a credit or a debit transaction
43
44     // signal that the balance exceed the maximum
45     final static short SW_EXCEED_MAXIMUM_BALANCE = 0x6A84;
46     // signal the the balance becomes negative
47     final static short SW_NEGATIVE_BALANCE = 0x6A85;
48
49
50     /* variables */
51     OwnerPIN pin;
52     static short balance ;
53 }
```

### I-2- Implémentations des méthodes nécessaires:

+ constructeurs:

```
private BankAdmin(byte[] bArray, int i, int j) {
    pin = new OwnerPIN(MAX_ERROR_PIN, MAX_PIN_LENGTH);

    // Initialization parametre pin
    pin.update(INIT_PIN, (short) 0, (byte) 0x04);
}
```

+ méthode install:

```
public static void install(byte bArray[], short bOffset, byte bLength) throws IOException {
    new BankAdmin(bArray, bOffset, bLength).register();
}
```

+méthode select:

```
public boolean select() {  
  
    // pas de selection si le pin est bloquer  
    if (pin.getTriesRemaining() == 0)  
        return false;  
  
    return true;  
}
```

+méthode deselect:

```
public void deselect() {  
    pin.reset();  
}
```

+méthode process:

```
public void process(APDU apdu) {  
    // Buffer=Objet APDU porte un tableau tampon de byte qui transfère  
    // l'entete + data entre la carte et le CAD  
    // du APDU entrant et sortant  
  
    byte[] buffer = apdu.getBuffer();  
  
    // exception qui teste sur la commande de selection  
    if (apdu.isISOInterindustryCLA()) {  
        if (buffer[ISO7816.OFFSET_INS] == (byte) (0xA4)) {  
            return;  
        } else {  
            ISOException.throwIt(ISO7816.SW_CLA_NOT_SUPPORTED);  
        }  
    }  
    //if (this.selectingApplet()) return;  
  
    // Vérifier si réinitialisation a une CLA correcte qui spécifie la  
    // structure de commandement  
    if (this.selectingApplet())  
        return;  
    if (buffer[ISO7816.OFFSET_CLA] != CLA_MONAPPLET) {  
        ISOException.throwIt(ISO7816.SW_CLA_NOT_SUPPORTED);  
    }  
  
    switch (buffer[ISO7816.OFFSET_INS]) {  
        case INS_TEST_CODE_PIN:  
            verify(apdu);  
            break;  
        case INS_INCREMENTER_COMPTE:  
            credit(apdu);  
            break;  
        case INS_DECREMENTER_COMPTE:  
            debit(apdu);  
            break;  
        case INS_INTERROGER_COMPTE:  
            getBalance(apdu);  
            break;  
        default:  
            ISOException.throwIt(ISO7816.SW_INS_NOT_SUPPORTED);  
    }  
}
```

+méthode credit:

```
private void credit(APDU apdu) {

    byte[] buffer = apdu.getBuffer();

    // LC byte denotes the number of bytes in the
    // data field of the command APDU
    byte numBytes = buffer[ISO7816.OFFSET_LC];

    // indicate that this APDU has incoming data
    // and receive data starting from the offset
    // ISO7816.OFFSET_CDATA following the 5 header
    // bytes.
    byte byteRead = (byte) (apdu.setIncomingAndReceive());

    // it is an error if the number of data bytes
    // read does not match the number in LC byte
    if ((numBytes != 1) || (byteRead != 1))
        ISOException.throwIt(ISO7816.SW_WRONG_LENGTH);

    // get the credit amount
    byte creditAmount = buffer[ISO7816.OFFSET_CDATA];

    // check the new balance
    if ((short) (balance + creditAmount) > MAX_BALANCE)
        ISOException.throwIt(SW_EXCEED_MAXIMUM_BALANCE);

    // credit the amount
    balance = (short) (balance + creditAmount);

}
```

+méthode debit:

```
private void debit(APDU apdu) {

    byte[] buffer = apdu.getBuffer();

    byte numBytes = (byte) (buffer[ISO7816.OFFSET_LC]);

    byte byteRead = (byte) (apdu.setIncomingAndReceive());

    if ((numBytes != 1) || (byteRead != 1))
        ISOException.throwIt(ISO7816.SW_WRONG_LENGTH);

    // get debit amount
    byte debitAmount = buffer[ISO7816.OFFSET_CDATA];

    // check the new balance
    if ((short) (balance - debitAmount) < (short) 0)
        ISOException.throwIt(SW_NEGATIVE_BALANCE);

    balance = (short) (balance - debitAmount);

}
```

+méthode getbalance:

```
private void getBalance(APDU apdu) {
    byte[] buffer = apdu.getBuffer();

    short le = apdu.setOutgoing();

    if (le < 2)
        ISOException.throwIt(ISO7816.SW_WRONG_LENGTH);

    apdu.setOutgoingLength((byte) 2);

    buffer[0] = (byte) (balance >> 8);
    buffer[1] = (byte) (balance & 0xFF);

    Util.setShort(buffer, (short)0, balance);

    apdu.sendBytes((short) 0, (short) 2);
}
```

+méthode verify:

```
private void verify(APDU apdu) {
    byte[] buffer = apdu.getBuffer();
    // retrieve the PIN data for validation.
    byte byteRead = (byte) (apdu.setIncomingAndReceive());
    // check pin
    // the PIN data is read into the APDU buffer
    // at the offset ISO7816.OFFSET_CDATA
    // the PIN data length = byteRead
    if (pin.getTriesRemaining() <= (byte) 1) {
        if (pin.check(buffer, ISO7816.OFFSET_CDATA, byteRead) == false) {
            ISOException.throwIt(SW_EXCEED_TRY_LIMIT);
        }
    }
    if (pin.check(buffer, ISO7816.OFFSET_CDATA, byteRead) == false)
        ISOException.throwIt(SW_VERIFICATION_FAILED);
}
```

© 2023 krimowess™ Tous les droits réservés

